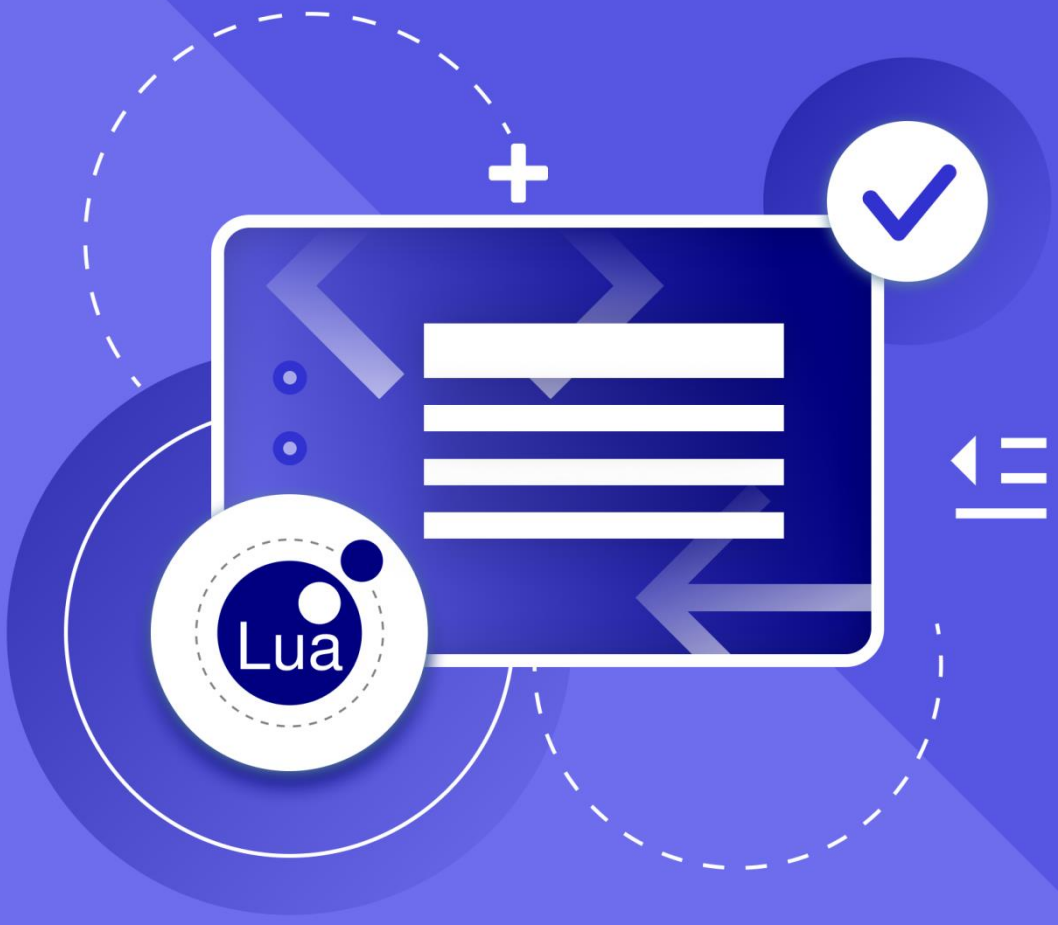


МойОфис

Приложение

Справочник макрокоманд на языке Lua

502220.29144487-2016-01 34

СОДЕРЖАНИЕ

1 Работа с макрокомандами 1048
1.1 Редактор макрокоманд 1049
1.1.1 Окно редактора макрокоманд 1049
1.1.2 Создание макрокоманд 1050
1.1.3 Выполнение макрокоманд 1050
1.1.4 Редактирование макрокоманд 1050
1.1.5 Удаление макрокоманд 1050
1.1.6 Отладка макрокоманд 1050
1.2 Пример подготовки и запуска макрокоманды 1053
1.2.1 Редактирование и запуск макрокоманды в текстовом документе 1053
1.2.2 Описание примера работы макрокоманды 1054
1.3 Преобразование макрокоманд на языке программирования VBA 1055
2 Объектная модель МойОфис 1055
3 Справочник функций DocumentAPI 1056
3.1 Диаграммы 1056
3.1.1 Таблица DocumentAPI.Charts 1057
3.1.1.1 Метод Charts:getChartsCount 1058
3.1.1.2 Метод Charts:getChart 1058
3.1.1.3 Метод Charts:getChartIndexByDrawingIndex 1059
3.1.2 Таблица DocumentAPI.Chart 1059
3.1.2.1 Метод Chart:getType 1059
3.1.2.2 Метод Chart:setType 1059
3.1.2.3 Метод Chart:getRangesCount 1059
3.1.2.4 Метод Chart:getRange 1059
3.1.2.5 Метод Chart:getTitle 1060
3.1.2.6 Метод Chart:setRange 1060
3.1.2.7 Метод Chart:setRect 1060
3.1.2.8 Метод Chart:isEmpty 1060
3.1.2.9 Метод Chart:isSolidRange 1060
3.1.2.10 Метод Chart:is3D 1061
3.1.2.11 Метод Chart:getDirectionType 1061
3.1.2.12 Метод Chart:getChartLabels 1061
3.1.2.13 Метод Chart:getRangeAsString 1061
3.1.2.14 Метод Chart:applySettings 1062
3.1.3 Таблица DocumentAPI.ChartLabelsDetectionMode 1062
3.1.4 Таблица DocumentAPI.ChartLabelsInfo 1063

3.1.5 Таблица DocumentAPI.ChartRangeInfo 1064
3.1.6 Таблица DocumentAPI.ChartRangeType 1064
3.1.7 Таблица DocumentAPI.ChartSeriesDirectionType 1065
3.1.8 Таблица DocumentAPI.ChartType 1065
3.2 Именованные выражения 1066
3.2.1 Таблица DocumentAPI.NamedExpressions 1067
3.2.1.1 Метод NamedExpressions:get 1067
3.2.1.2 Метод NamedExpressions:enumerate 1067
3.2.1.3 Метод NamedExpression:addExpression 1068
3.2.1.4 Метод NamedExpressions:removeExpression 1068
3.2.2 Таблица DocumentAPI.NamedExpression 1068
3.2.2.1 Метод NamedExpression:getName 1069
3.2.2.2 Метод NamedExpression:getExpression 1069
3.2.2.3 Метод NamedExpression:getCellRange 1069
3.2.3 Таблица DocumentAPI.NamedExpressionsValidationResult 1069
3.3 Макрокоманды 1069
3.3.1 Таблица DocumentAPI.Scripts 1070
3.3.1.1 Метод Scripts:getScript 1070
3.3.1.2 Метод Scripts:setScript 1070
3.3.1.3 Метод Scripts:removeScript 1071
3.3.1.4 Метод Scripts:enumerate 1071
3.3.2 Таблица DocumentAPI.Script 1071
3.3.2.1 Таблица DocumentAPI.Scripting 1071
3.3.2.1.1 Метод Scripting:runScript 1071
3.3.2.2 Метод Script:getName 1071
3.3.2.3 Метод Script:setName 1072
3.3.2.4 Метод Script:getBody 1072
3.3.2.5 Метод Script:setBody 1072
3.3.3 Функция DocumentAPI.createScripting 1072
3.4 Разделы (секции) документа 1073
3.4.1 Таблица DocumentAPI.Sections 1074
3.4.1.1 Метод Sections:enumerate 1074
3.4.2 Таблица DocumentAPI.Section 1074
3.4.2.1 Метод Section:setPageProperties 1074
3.4.2.2 Метод Section:getPageProperties 1074
3.4.2.3 Метод Section:setPageOrientation 1075
3.4.2.4 Метод Section:getPageOrientation 1075

3.4.2.5 Метод Section:getRange 1075
3.4.2.6 Метод Section:getHeaders 1075
3.4.2.7 Метод Section:getFooters 1076
3.4.3 Таблица DocumentAPI.HeadersFooters 1076
3.4.3.1 Метод HeadersFooters:enumerate 1076
3.4.4 Таблица DocumentAPI.HeaderFooter 1077
3.4.4.1 Метод HeaderFooter:getType 1077
3.4.4.2 Метод HeaderFooter:getBlocks 1077
3.4.4.3 Метод HeaderFooter:getRange 1077
3.4.5 Таблица DocumentAPI.HeaderFooterType 1078
3.4.6 Таблица DocumentAPI.PageOrientation 1078
3.4.7 Таблица DocumentAPI.Insets 1078
3.4.8 Таблица DocumentAPI.PageProperties 1079
3.4.8.1 Метод PageProperties:__eq 1080
3.4.8.2 Метод PageProperties:__ne 1080
3.5 Блоки, параграфы 1080
3.5.1 Таблица DocumentAPI.Blocks 1080
3.5.1.1 Метод Blocks:getBlock 1080
3.5.1.2 Метод Blocks:getParagraph 1081
3.5.1.3 Метод Blocks:getTable 1081
3.5.1.4 Метод Blocks:getShape 1081
3.5.1.5 Метод Blocks:getField 1081
3.5.1.6 Метод Blocks:enumerate 1081
3.5.1.7 Метод Blocks:enumerateParagraphs 1082
3.5.1.8 Метод Blocks:enumerateTables 1082
3.5.1.9 Метод Blocks:enumerateShapes 1082
3.5.1.10 Метод Blocks:enumerateFields 1082
3.5.2 Таблица DocumentAPI.Block 1082
3.5.2.1 Методы toParagraph, toTable, toShape, toField 1083
3.5.2.2 Метод Block.getRange 1083
3.5.2.3 Метод Block.remove 1083
3.5.2.4 Метод Block.getSection 1083
3.5.3 Таблица DocumentAPI.Field 1084
3.5.4 Таблица DocumentAPI.Paragraphs 1084
3.5.4.1 Метод Paragraphs:setListSchema 1084
3.5.4.2 Метод Paragraphs:setListLevel 1085
3.5.4.3 Метод Paragraphs:increaseListLevel 1085

3.5.4.4 Метод Paragraphs:decreaseListLevel 1085
3.5.4.5 Метод Paragraphs:enumerate 1085
3.5.5 Таблица DocumentAPI.Paragraph 1086
3.5.5.1 Метод Paragraph:getParagraphProperties 1086
3.5.5.2 Метод Paragraph:setParagraphProperties 1087
3.5.5.3 Метод Paragraph:getListSchema 1088
3.5.5.4 Метод Paragraph:setListSchema 1088
3.5.5.5 Метод Paragraph:getListLevel 1088
3.5.5.6 Метод Paragraph:setListLevel 1088
3.5.5.7 Метод Paragraph:increaseListLevel 1089
3.5.5.8 Метод Paragraph:decreaseListLevel 1089
3.5.6 Таблица DocumentAPI.ListSchema 1089
3.5.7 Таблица DocumentAPI.ParagraphProperties 1090
3.5.8 Таблица DocumentAPI.LineSpacing 1094
3.5.9 Таблица DocumentAPI.LineSpacingRule 1094
3.5.10 Таблица DocumentAPI.Alignment 1096
3.6 Рецензирование документов 1097
3.6.1 Таблица DocumentAPI.TrackedChange 1098
3.6.1.1 Метод TrackedChange:getRange 1098
3.6.1.2 Метод TrackedChange:getType 1099
3.6.1.3 Метод TrackedChange:getInfo 1099
3.6.2 Таблица DocumentAPI.TrackedChangeInfo 1099
3.6.2.1 Метод TrackedChangeInfo:__eq 1100
3.6.2.2 Метод TrackedChangeInfo:__ne 1100
3.6.3 Таблица DocumentAPI.DateTime 1100
3.6.3.1 Метод DateTime:__eq 1100
3.6.3.2 Метод DateTime:__ne 1100
3.6.4 Таблица DocumentAPI.TrackedChangeType 1100
3.6.5 Таблица DocumentAPI.Comments 1101
3.6.5.1 Метод Comments:enumerate 1101
3.6.6 Таблица DocumentAPI.Comment 1102
3.6.6.1 Метод Comment:getRange 1102
3.6.6.2 Метод Comment:getText 1102
3.6.6.3 Метод Comment:getInfo 1102
3.6.6.4 Метод Comment:isResolved 1103
3.6.6.5 Метод Comment:getReplies 1103
3.7 Работа с закладками 1103

3.7.1 Таблица DocumentAPI.Bookmarks 1104
3.7.1.1 Метод Bookmarks:getBookmarkRange 1105
3.7.1.2 Метод Bookmarks:removeBookmark 1105
3.8 Таблицы и ячейки 1105
3.8.1 Доступ к таблицам 1105
3.8.2 Доступ к ячейкам 1106
3.8.2.1 Таблица DocumentAPI.CellFormat 1109
3.8.2.2 Таблица DocumentAPI.AccountingCellFormatting 1111
3.8.2.3 Таблица DocumentAPI.PercentageCellFormatting 1112
3.8.2.4 Таблица DocumentAPI.NumberCellFormatting 1113
3.8.2.5 Таблица DocumentAPI.CurrencyCellFormatting 1113
3.8.2.6 Таблица DocumentAPI.CurrencySignPlacement 1115
3.8.2.7 Таблица DocumentAPI.DateTimeCellFormatting 1115
3.8.2.8 Таблица DocumentAPI.DatePatterns 1116
3.8.2.9 Таблица DocumentAPI.TimePatterns 1116
3.8.2.10 Таблица DocumentAPI.FractionCellFormatting 1116
3.8.2.11 Таблица DocumentAPI.ScientificCellFormatting 1117
3.8.3 Форматирование ячеек 1118
3.8.4 Форматирование границ ячеек 1119
3.8.4.1 Таблица DocumentAPI.Borders 1120
3.8.4.2 Таблица DocumentAPI.RangeBorders 1122
3.8.4.3 Таблица DocumentAPI.LineProperties 1122
3.8.4.3.1 Поле LineProperties.style 1123
3.8.4.3.2 Поле LineProperties.width 1123
3.8.4.3.3 Поле LineProperties.color 1123
3.8.4.3.4 Поле LineProperties.headLineEndingProperties 1123
3.8.4.3.5 Поле LineProperties.tailLineEndingProperties 1123
3.8.4.4 Таблица DocumentAPI.LineEndingProperties 1123
3.8.4.5 Таблица DocumentAPI.LineStyle 1124
3.8.4.6 Таблица DocumentAPI.LineEndingStyle 1125
3.8.5 Объединение и разделение ячеек таблицы 1126
3.8.6 Таблица DocumentAPI.Table 1126
3.8.6.1 Метод Table:setName 1127
3.8.6.2 Метод Table:getName 1127
3.8.6.3 Метод Table:getRowsCount 1127
3.8.6.4 Метод Table:getColumnsCount 1128
3.8.6.5 Метод Table:getCell 1128

3.8.6.6 Метод Table:getCellRange 1128
3.8.6.7 Метод Table:insertColumnAfter 1128
3.8.6.8 Метод Table:insertColumnBefore 1129
3.8.6.9 Метод Table:insertRowAfter 1130
3.8.6.10 Метод Table:insertRowBefore 1131
3.8.6.11 Метод Table:removeColumn 1131
3.8.6.12 Метод Table:removeRow 1131
3.8.6.13 Метод Table:setColumnWidth 1132
3.8.6.14 Метод Table:setRowHeight 1132
3.8.6.15 Метод Table:duplicate 1133
3.8.6.16 Метод Table:remove 1133
3.8.6.17 Метод Table:moveTo 1133
3.8.6.18 Метод Table:setShowZeroValue 1133
3.8.6.19 Метод Table:getShowZeroValue 1134
3.8.6.20 Метод Table:setVisible 1134
3.8.6.21 Метод Table:isVisible 1134
3.8.6.22 Метод Table:getFrozenRange 1134
3.8.6.23 Метод Table:freeze 1135
3.8.6.24 Метод Table:__eq 1135
3.8.6.25 Метод Table:__ne 1135
3.8.6.26 Метод Table:setPrintArea 1135
3.8.6.27 Метод Table:setPrintAreas 1136
3.8.6.28 Метод Table:getPrintAreas 1136
3.8.6.29 Метод Table:getCharts 1136
3.8.6.30 Метод Table:getImages 1137
3.8.6.31 Метод Table:getMediaObjects 1137
3.8.6.32 Метод Table:getNamedExpressions 1137
3.8.6.33 Группировка строк и колонок таблицы 1137
3.8.6.34 Управление видимостью строк / колонок 1138
3.8.7 Таблица DocumentAPI.Cell 1138
3.8.7.1 Метод Cell:getRange 1139
3.8.7.2 Метод Cell:setBorders 1139
3.8.7.3 Метод Cell:setFormula 1139
3.8.7.4 Метод Cell:getFormat 1139
3.8.7.5 Метод Cell:setFormat 1139
3.8.7.6 Метод Cell:getFormattedValue 1142
3.8.7.7 Метод Cell:setFormattedValue 1142

3.8.7.8 Метод Cell:unmerge 1142
3.8.7.9 Метод Cell:getHyperlink 1142
3.8.7.10 Метод Cell:setContent 1143
3.8.7.11 Метод Cell:getBorders 1143
3.8.7.12 Метод Cell:getRawValue 1143
3.8.7.13 Метод Cell:getCustomFormat 1143
3.8.7.14 Метод Cell:setCustomFormat 1143
3.8.7.15 Метод Cell:setBool 1143
3.8.7.16 Метод Cell:setNumber 1144
3.8.7.17 Метод Cell:setText 1144
3.8.7.18 Метод Cell:getFormulaAsString 1144
3.8.7.19 Метод Cell:getCellProperties 1144
3.8.7.20 Метод Cell:setCellProperties 1144
3.8.7.21 Метод Cell:getParagraphProperties 1145
3.8.7.22 Метод Cell:setParagraphProperties 1145
3.8.7.23 Метод Cell:getPivotTable 1145
3.8.8 Таблица DocumentAPI.CellProperties 1146
3.8.9 Таблица DocumentAPI.VerticalAlignment 1147
3.8.10 Таблица DocumentAPI.Hyperlink 1148
3.8.10.1 Метод Hyperlink:__eq 1148
3.8.10.2 Метод Hyperlink:__ne 1148
3.8.11 Таблица DocumentAPI.TextLayout 1148
3.8.12 Таблица DocumentAPI.TextOrientation 1149
3.8.12.1 Метод TextOrientation:getAngle 1149
3.8.13 Таблица DocumentAPI.CellPosition 1150
3.8.13.1 Поле CellPosition.column 1150
3.8.13.2 Поле CellPosition.row 1150
3.8.13.3 Метод CellPosition:toString 1150
3.8.13.4 Метод CellPosition:__eq 1151
3.8.13.5 Метод CellPosition:__ne 1151
3.8.14 Таблица DocumentAPI.CellRange 1151
3.8.14.1 Метод CellRange:enumerate 1151
3.8.14.2 Метод CellRange:getBeginRow 1151
3.8.14.3 Метод CellRange:getBeginColumn 1152
3.8.14.4 Метод CellRange:getLastRow 1152
3.8.14.5 Метод CellRange:getLastColumn 1152
3.8.14.6 Метод CellRange:setBorders 1152

3.8.14.7 Метод CellRange:insertCurrentDateTime 1153
3.8.14.8 Метод CellRange:getCellProperties 1153
3.8.14.9 Метод CellRange:setCellProperties 1153
3.8.14.10 Метод CellRange:merge 1154
3.8.14.11 Метод CellRange:unmerge 1154
3.8.15 Таблица DocumentApi.DateTimeFormat 1154
3.8.16 Таблица DocumentAPI.CellRangePosition 1154
3.8.16.1 Метод CellRangePosition:toString 1155
3.8.16.2 Метод CellRangePosition:__eq 1155
3.8.16.3 Метод CellRangePosition:__ne 1155
3.8.17 Таблица DocumentAPI.FrozenRangePosition 1156
3.8.17.1 Конструкторы 1156
3.8.17.2 Метод FrozenRangePosition:create 1156
3.8.17.3 Метод FrozenRangePosition:createFrozenArea 1156
3.8.17.4 Метод FrozenRangePosition:createFrozenRows 1157
3.8.17.5 Метод FrozenRangePosition:createFrozenCols 1157
3.8.17.6 Метод FrozenRangePosition:isRowsCols 1157
3.8.17.7 Метод FrozenRangePosition:isArea 1157
3.8.17.8 Метод FrozenRangePosition:isRows 1158
3.8.17.9 Метод FrozenRangePosition:isCols 1158
3.8.17.10 Метод FrozenRangePosition:__eq 1158
3.8.17.11 Метод FrozenRangePosition:__ne 1158
3.8.18 Таблица DocumentAPI.TableRangeInfo 1158
3.9 Сводные таблицы 1159
3.9.1 Таблица DocumentAPI.PivotTablesManager 1159
3.9.1.1 Метод PivotTablesManager:create 1160
3.9.2 Таблица DocumentAPI.PivotTable 1160
3.9.2.1 Метод PivotTable:remove 1160
3.9.2.2 Метод PivotTable:getSourceRangeAddress 1160
3.9.2.3 Метод PivotTable:getSourceRange 1160
3.9.2.4 Метод PivotTable:getPivotRange 1161
3.9.2.5 Метод PivotTable:changeSourceRange 1161
3.9.2.6 Метод PivotTable:isRowGrandTotalEnabled 1161
3.9.2.7 Метод PivotTable:isColumnGrandTotalEnabled 1161
3.9.2.8 Метод PivotTable:getPivotTableCaptions 1161
3.9.2.9 Метод PivotTable:getPivotTableLayoutSettings 1162
3.9.2.10 Метод PivotTable:getUnsupportedFeatures 1162

3.9.2.11 Метод PivotTable:getFieldsList ... 1162
3.9.2.12 Метод PivotTable:getRowFields ... 1163
3.9.2.13 Метод PivotTable:getColumnFields ... 1163
3.9.2.14 Метод PivotTable:getValueFields ... 1163
3.9.2.15 Метод PivotTable:getPageFields ... 1163
3.9.2.16 Метод PivotTable:getFieldCategories ... 1164
3.9.2.17 Метод PivotTable:getFieldItems ... 1164
3.9.2.18 Метод PivotTable:getFieldItemsByName ... 1164
3.9.2.19 Метод PivotTable:getFilter ... 1164
3.9.2.20 Метод PivotTable:getFilters ... 1164
3.9.2.21 Метод PivotTable:update ... 1165
3.9.2.22 Метод PivotTable:createPivotTableEditor ... 1165
3.9.3 Таблица DocumentAPI.PivotTableCaptions ... 1165
3.9.4 Таблица DocumentAPI.PivotTableLayoutSettings ... 1166
3.9.5 Таблица DocumentAPI.PivotTableReportLayout ... 1166
3.9.6 Таблица DocumentAPI.ValueFieldsOrientation ... 1167
3.9.7 Таблица DocumentAPI.PageFieldOrder ... 1167
3.9.8 Таблица DocumentAPI.PivotTableUnsupportedFeature ... 1167
3.9.9 Таблица DocumentAPI.PivotTableFieldCategories ... 1168
3.9.9.1 Метод PivotTableFieldCategories:enumerate ... 1168
3.9.10 Таблица DocumentAPI.PivotTableFunction ... 1168
3.9.11 Таблица DocumentAPI.PivotTableFilters ... 1169
3.9.11.1 Метод PivotTableFilters:enumerate ... 1169
3.9.12 Таблица DocumentAPI.PivotTableFilter ... 1169
3.9.12.1 Метод PivotTableFilter:getFieldName ... 1170
3.9.12.2 Метод PivotTableFilter:getCount ... 1170
3.9.12.3 Метод PivotTableFilter:getName ... 1170
3.9.12.4 Метод PivotTableFilter:isHidden ... 1171
3.9.12.5 Метод PivotTableFilter:setHidden ... 1171
3.9.13 Таблица DocumentAPI.PivotTableField ... 1171
3.9.14 Таблица DocumentAPI.PivotTableFieldProperties ... 1171
3.9.15 Таблица DocumentAPI.PivotTableCategoryField ... 1172
3.9.16 Таблица DocumentAPI.PivotTableValueField ... 1172
3.9.17 Таблица DocumentAPI.PivotTablePageField ... 1173
3.9.18 Таблица DocumentAPI.PivotTableItems ... 1173
3.9.18.1 Метод PivotTableItems:enumerate ... 1173
3.9.19 Таблица DocumentAPI.PivotTableItem ... 1173

3.9.19.1 Метод PivotTableItem:getName 1174
3.9.19.2 Метод PivotTableItem:getAlias 1174
3.9.19.3 Метод PivotTableItem:getItemType 1174
3.9.19.4 Метод PivotTableItem:isCollapsed 1174
3.9.20 Таблица DocumentAPI.PivotTableItemType 1174
3.9.21 Таблица DocumentAPI.PivotTableEditor 1175
3.9.21.1 Метод PivotTableEditor:addField 1175
3.9.21.2 Метод PivotTableEditor:moveField 1176
3.9.21.3 Метод PivotTableEditor:removeField 1176
3.9.21.4 Метод PivotTableEditor:reorderField 1176
3.9.21.5 Метод PivotTableEditor:enableField 1177
3.9.21.6 Метод PivotTableEditor:disableField 1177
3.9.21.7 Метод PivotTableEditor:setSummarizeFunction 1177
3.9.21.8 Метод PivotTableEditor:setFilter 1178
3.9.21.9 Метод PivotTableEditor:setFilters 1178
3.9.21.10 Метод PivotTableEditor:setCaptions 1178
3.9.21.11 Метод PivotTableEditor:setLayoutSettings 1179
3.9.21.12 Метод PivotTableEditor:setGrandTotalSettings 1179
3.9.21.13 Метод PivotTableEditor:apply 1179
3.9.22 Таблица DocumentAPI.PivotTableUpdateResult 1180
3.9.23 Таблица DocumentAPI.PivotTableFieldCategory 1181
3.10 Графические объекты 1181
3.10.1 Таблица DocumentAPI.MediaObjects 1182
3.10.1.1 Метод MediaObjects:enumerate 1183
3.10.2 Таблица DocumentAPI.MediaObject 1183
3.10.2.1 Метод MediaObject:toImage 1184
3.10.2.2 Метод MediaObject:getFrame 1184
3.10.3 Таблица DocumentAPI.Image 1185
3.10.3.1 Метод Image:getFrame 1185
3.10.3.2 Метод Image:remove 1186
3.10.4 Таблица DocumentAPI.Images 1186
3.10.4.1 Метод Images:enumerate 1186
3.10.5 Таблица DocumentAPI.AbsoluteFrame 1187
3.10.5.1 Метод AbsoluteFrame:moveTo 1187
3.10.5.2 Метод AbsoluteFrame:getTopLeft 1188
3.10.5.3 Метод AbsoluteFrame:scale 1188
3.10.5.4 Метод AbsoluteFrame:setDimensions 1188

3.10.5.5 Метод AbsoluteFrame:getDimensions ... 1189
3.10.6 Таблица DocumentAPI.ScaleFrom ... 1189
3.10.7 Таблица DocumentAPI.InlineFrame ... 1190
3.10.7.1 Метод InlineFrame:setPosition ... 1191
3.10.7.2 Метод InlineFrame:getPosition ... 1191
3.10.7.3 Метод InlineFrame:setDimensions ... 1192
3.10.7.4 Метод InlineFrame:getDimensions ... 1192
3.10.7.5 Метод InlineFrame:setWrapType ... 1192
3.10.7.6 Метод InlineFrame:getWrapType ... 1192
3.10.8 Таблица DocumentAPI.TextWrapType ... 1193
3.10.9 Таблица DocumentAPI.TextAnchoredPosition ... 1193
3.10.9.1 Метод TextAnchoredPosition:__eq ... 1193
3.10.9.2 Метод TextAnchoredPosition:__ne ... 1193
3.10.10 Таблица DocumentAPI.HorizontalTextAnchoredPosition ... 1194
3.10.10.1 Метод HorizontalTextAnchoredPosition:__eq ... 1194
3.10.10.2 Метод HorizontalTextAnchoredPosition:__ne ... 1194
3.10.11 Таблица DocumentAPI.VerticalTextAnchoredPosition ... 1194
3.10.11.1 Метод VerticalTextAnchoredPosition:__eq ... 1195
3.10.11.2 Метод VerticalTextAnchoredPosition:__ne ... 1195
3.10.12 Таблица DocumentAPI.VerticalRelativeTo ... 1195
3.10.13 Таблица DocumentAPI.HorizontalRelativeTo ... 1196
3.10.14 Таблица DocumentAPI.VerticalAnchorAlignment ... 1196
3.10.15 Таблица DocumentAPI.HorizontalAnchorAlignment ... 1197
3.10.16 Таблица DocumentAPI.Shape ... 1197
3.10.16.1 Метод Shape:getShapeProperties ... 1197
3.10.16.2 Метод Shape:setShapeProperties ... 1197
3.10.17 Таблица DocumentAPI.ShapeProperties ... 1197
3.10.17.1 Поле ShapeProperties:borderProperties ... 1198
3.10.17.2 Поле ShapeProperties:verticalAlignment ... 1198
3.10.17.3 Поле ShapeProperties:fill ... 1198
3.10.17.4 Поле ShapeProperties:shapeTextLayout ... 1198
3.10.18 Таблица DocumentAPI.ShapeTextLayout ... 1198
3.10.19 Таблица DocumentAPI.Fill ... 1199
3.10.19.1 Метод Fill:getColor ... 1199
3.10.19.2 Метод Fill:getUrl ... 1199
3.10.19.3 Метод Fill:isNoFill ... 1199
3.11 Поиск в документе ... 1199

3.11.1 Метод DocumentAPI:createSearch 1199
3.11.2 Таблица DocumentAPI.Search 1199
3.11.2.1 Метод Search:findText 1200
3.12 Загрузка, сохранение, экспорт, импорт документов 1200
3.12.1 Таблица DocumentAPI.FormulaType 1201
3.12.2 Таблица DocumentAPI.ExportFormat 1201
3.12.3 Таблица DocumentAPI.DocumentFormat 1201
3.12.4 Таблица DocumentAPI.DocumentSettings 1202
3.12.5 Таблица DocumentAPI.DocumentType 1202
3.12.6 Таблица DocumentAPI.SaveDocumentSettings 1203
3.12.7 Таблица DocumentAPI.LoadDocumentSettings 1203
3.12.8 Таблица DocumentAPI.Encoding 1204
3.12.9 Таблица DocumentAPI.TextExportSettings 1205
3.12.10 Таблица DocumentAPI.WorkbookExportSettings 1205
3.12.11 Таблица DocumentAPI.PrintingScope 1206
3.12.11.1 Метод PrintingScope:getCellRange 1206
3.12.11.2 Метод PrintingScope:usePrintArea 1206
3.12.12 Таблица DocumentAPI.PrintingScope.Type 1206
3.12.13 Таблица DocumentAPI.UserInfo 1207
3.12.14 Таблица DocumentAPI.PageNumbers 1207
3.12.14.1 Метод PageNumbers:contains 1208
3.12.14.2 Метод PageNumbers:getLast 1208
3.12.15 Таблица DocumentAPI.PageParity 1208
3.12.16 Таблица DocumentAPI.TimeZone 1208
3.12.17 Таблица DocumentAPI.DSVSettings 1209
3.12.18 Таблица DocumentAPI.LocaleInfo 1209
3.13 Таблица DocumentAPI.Application 1210
3.13.1 Метод application:createDocument 1210
3.13.2 Метод application:loadDocument 1211
3.13.3 Метод application:getMessenger 1211
3.14 Таблица DocumentAPI.document 1211
3.14.1 Метод document:saveAs 1211
3.14.2 Метод document:exportAs 1212
3.14.3 Метод document:merge 1213
3.14.4 Метод document:getBlocks 1213
3.14.5 Метод document:getBookmarks 1213
3.14.6 Метод document:getScripts 1213

3.14.7 Метод document:getRange 1213
3.14.8 Метод document:isChangesTrackingEnabled 1214
3.14.9 Метод document:setChangesTrackingEnabled 1214
3.14.10 Метод document:getComments 1214
3.14.11 Метод document:setPageProperties 1215
3.14.12 Метод document:setFormulaType 1215
3.14.13 Метод document:getFormulaType 1215
3.14.14 Метод document:setPageOrientation 1215
3.14.15 Метод document:enumerateSections 1215
3.14.16 Метод document:getSections 1215
3.14.17 Метод document:setMirroredMarginsEnabled 1216
3.14.18 Метод document:areMirroredMarginsEnabled 1216
3.14.19 Метод document:getPivotTablesManager 1216
3.14.20 Метод document:getNamedExpressions 1216
3.15 Таблица DocumentAPI.SaveUnsupportedFeature 1216
3.16 Таблица DocumentAPI.ColorRGBA 1217
3.16.1 Метод ColorRGBA:__eq 1217
3.16.2 Метод ColorRGBA:__ne 1217
3.17 Таблица DocumentAPI.TextProperties 1217
3.18 Таблица DocumentAPI.ScriptPosition 1219
3.19 Таблица DocumentAPI.Range 1219
3.19.1 Метод Range:getBegin 1221
3.19.2 Метод Range:getEnd 1221
3.19.3 Метод Range:extractText 1222
3.19.4 Метод Range:removeContent 1222
3.19.5 Метод Range:lockContent 1222
3.19.6 Метод Range:unlockContent 1223
3.19.7 Метод Range:isContentLocked 1223
3.19.8 Метод Range:replaceText 1224
3.19.9 Метод Range:setHyperlink 1224
3.19.10 Метод Range:getTextProperties 1225
3.19.11 Метод Range:setTextProperties 1225
3.19.12 Метод Range:enumerateBlocks 1225
3.19.13 Метод Range:enumerateTrackedChanges 1226
3.19.14 Метод Range:getComments 1226
3.19.15 Метод Range:getParagraphs 1226
3.19.16 Метод Range:getImages 1227

3.19.17 Метод Range:getInlineObjects 1227
3.20 Таблица DocumentAPI.Position 1228
3.20.1 Метод Position:insertText 1228
3.20.2 Метод Position:insertTable 1228
3.20.3 Метод Position:insertPageBreak 1228
3.20.4 Метод Position:insertLineBreak 1229
3.20.5 Метод Position:insertBookmark 1229
3.20.6 Метод Position:insertSectionBreak 1229
3.20.7 Метод Position:insertHyperlink 1229
3.20.8 Метод Position:insertImage 1229
3.20.9 Метод Position:removeBackward 1230
3.20.10 Метод Position:removeForward 1230
3.20.11 Метод Position:__eq 1230
3.20.12 Метод Position:__ne 1230
3.21 Таблица DocumentAPI.Messenger 1230
3.21.1 Метод Messenger:subscribe 1230
3.21.2 Метод Messenger:notify 1230
3.22 Таблица DocumentAPI.Connection 1231
3.23 Таблица DocumentAPI.Message 1231
3.23.1 Таблица Message.Severity 1231
3.23.2 Метод Message:__eq 1231
3.23.3 Метод Message:__ne 1231
3.23.4 Метод Message:getSeverity 1231
3.23.5 Метод Message:getText 1231
3.23.6 Метод Message:makeInfo 1231
3.23.7 Метод Message:makeWarning 1231
3.23.8 Метод Message:makeError 1232
3.24 Таблица DocumentAPI.Color 1232
3.24.1 Метод Color:getRGBAColor 1232
3.24.2 Метод Color:getThemeColorID 1232
3.24.3 Метод Color:__eq 1232
3.24.4 Метод Color:__ne 1232
3.25 Таблица DocumentAPI.ThemeColorID 1233
3.26 Таблица DocumentAPI.PrintSettings 1233
3.27 Таблица DocumentAPI.WorksheetPrinterFitType 1235
3.28 Таблица DocumentAPI.PrintDocumentResult 1235
3.29 Таблица DocumentAPI.SizeU 1236

3.29.1 Метод SizeU:toString 1236
3.30 Таблица DocumentAPI.PointU 1236
3.30.1 Метод PointU:toString 1237
3.31 Таблица DocumentAPI.RectU 1237
3.31.1 Метод RectU:toString 1237
3.32 Таблица DocumentAPI.VectorUInt 1238
3.32.1 Метод VectorUInt:size 1238
3.32.2 Метод VectorUInt:max_size 1238
3.32.3 Метод VectorUInt:empty 1238
3.32.4 Метод VectorUInt:clear 1239
3.32.5 Метод VectorUInt:push_back 1239
3.32.6 Метод VectorUInt:pop_back 1239
3.32.7 Метод VectorUInt:front 1239
3.32.8 Метод VectorUInt:back 1239
3.32.9 Метод VectorUInt:__getitem 1240
3.32.10 Метод VectorUInt:__setitem 1240
3.33 Таблица DocumentAPI.VectorString 1240
3.33.1 Метод VectorString:size 1240
3.33.2 Метод VectorString:max_size 1241
3.33.3 Метод VectorString:empty 1241
3.33.4 Метод VectorString:clear 1241
3.33.5 Метод VectorString:push_back 1241
3.33.6 Метод VectorString:pop_back 1242
3.33.7 Метод VectorString:front 1242
3.33.8 Метод VectorString:back 1242
3.33.9 Метод VectorString:__getitem 1242
3.33.10 Метод VectorString:__setitem 1242
4 Справочник функций EditorAPI 1243
4.1 Функция EditorAPI.getSelection 1243
4.2 Функция EditorAPI.setSelection 1243
4.3 Функция EditorAPI.messageBox 1244
4.4 Функция EditorAPI.showPrintDialog 1244
4.5 Функция EditorAPI.printDocument 1245
4.6 Функция EditorAPI.isPrinterAvailable 1245
5 Функции для работы со строками в формате Юникод (UTF-8) 1245
5.1 Функция utf8.upper 1245
5.2 Функция utf8.lower 1246

5.3 Функция utf8.substr 1246
5.4 Функция utf8.compare 1246
5.5 Функция utf8.islower 1247
5.6 Функция utf8.isupper 1247
5.7 Функция utf8.isdigit 1247
5.8 Функция utf8.isalpha 1248
5.9 Функция utf8.next 1248
6 Функции для работы с регулярными выражениями 1248
6.1 Функция Re.create 1248
6.2 Функция Re.match 1249
6.2.1 Флаги, используемые в Re.match 1249
6.3 Функция Re.search 1251
6.4 Функция Re.replace 1251
6.5 Флаги, используемые для замены 1252
7 Класс Matches 1252
7.1 Метод getFirst 1252
7.2 Метод getLength 1253
7.3 Метод getSize 1253
7.4 Метод getString 1253
7.5 Метод _tostring 1254

1 РАБОТА С МАКРОКОМАНДАМИ

В данном разделе описаны действия по созданию, выполнению и отладке макрокоманд в редакторе документов МойОфис.

1.1 Редактор макрокоманд

1.1.1 Окно редактора макрокоманд

Для работы с макрокомандами используется редактор макрокоманд. Чтобы открыть окно редактора, в приложении «МойОфис Текст» или «МойОфис Таблица» выберите пункт командного меню **Инструменты > Макрокоманды > Редактор макрокоманд**.

Окно редактора макрокоманд содержит (см. Рисунок 1108):

1. Список макрокоманд документа.
2. Кнопки для создания + и удаления − макрокоманд.
3. Область ввода текста макрокоманд.
4. Кнопки выполнения (см. раздел Выполнение макрокоманд) и отладки (см. раздел Отладка макрокоманд) макрокоманд. Кнопки становятся активными, изменяя цвет, после ввода текста макрокоманд в области **3** (см. раздел Редактирование макрокоманд).
5. Область вывода результата выполнения макрокоманд, а также отображения информации в процессе отладки макрокоманд.

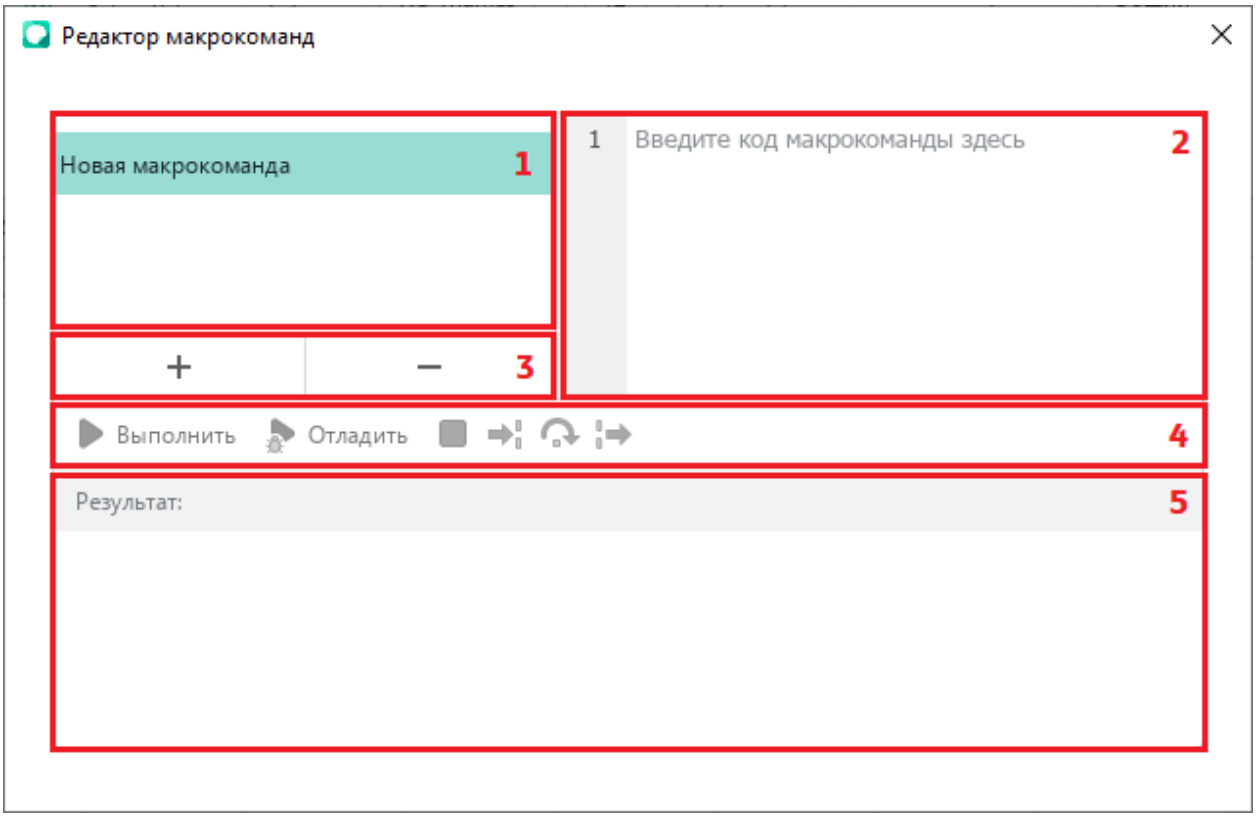


Рисунок 1108 – Окно редактора макрокоманд

1.1.2 Создание макрокоманд

Для создания макрокоманды выполните следующие действия (см. Рисунок 1108):

1. Нажмите кнопку + в области **2**.
2. Введите наименование макрокоманды в соответствующей строке перечня макрокоманд в области **1**. Чтобы сохранить название, нажмите клавишу **Enter** или щелкните мышью по любой области окна **Редактор макрокоманд.**
3. Введите текст макрокоманды в области ввода **3**.

1.1.3 Выполнение макрокоманд

Чтобы выполнить макрокоманду, выберите ее в перечне макрокоманд и нажмите кнопку **Выполнить** (см. Рисунок 1108).

Результат выполнения макрокоманды отображается в области **5**.

1.1.4 Редактирование макрокоманд

Чтобы редактировать макрокоманду, выберите ее в перечне макрокоманд и внесите необходимые изменения в ее текст в области **3** (см. Рисунок 1108).

1.1.5 Удаление макрокоманд

Чтобы удалить макрокоманду, выберите ее в перечне макрокоманд и нажмите кнопку − в области **2** (см. Рисунок 1108).

1.1.6 Отладка макрокоманд

Для отладки макрокоманды выполните следующие действия:

1. Выберите наименование макрокоманды в перечне макрокоманд.
2. Установите (при необходимости) в тексте макрокоманд точки останова отладчика, щелкнув мышью справа от номера строки макрокоманды. Строка точки останова будет отмечена значком . Для удаления точки останова щелкните мышью на значок .
3. Нажмите кнопку **Отладить**. Запустится режим отладки и окно редактора макрокоманд изменит свой вид (см. Рисунок 1109).

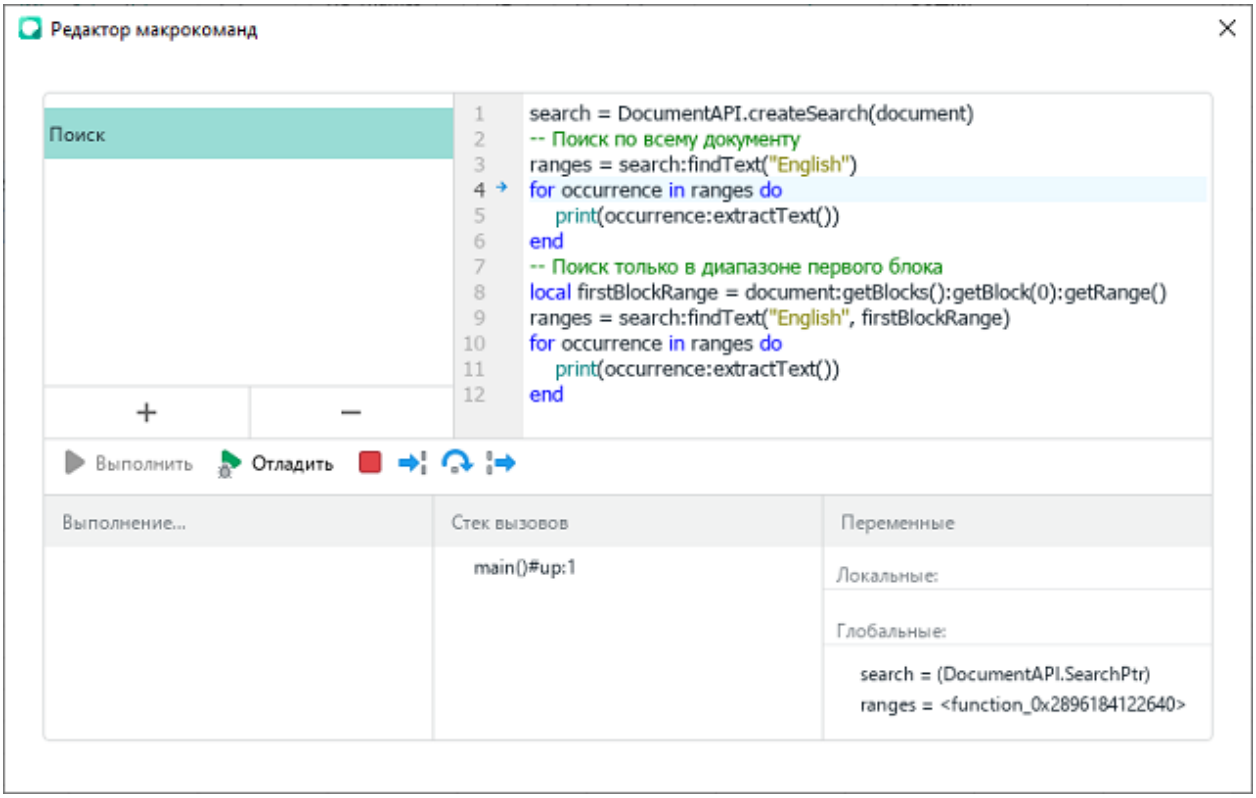


Рисунок 1109 – Окно редактора макрокоманд в начале отладки

Процесс отладки макрокоманды остановится на первой точке останова. Если точки останова отсутствуют, то отладка начнется с остановки на первой строке макрокоманды.

Для продолжения отладки нажмите одну из следующих кнопок: (см. Рисунок 1109):

– – для выполнения одного шага отладки или захода в тело функции, если таковая есть в текущей позиции отладки;
– – для выполнения одного шага отладки без захода в тело функции;
– – для продолжения выполнения макрокоманды до момента выхода из функции, в которой отладчик находится в текущей позиции.

Для прерывания процесса отладки нажмите кнопку (см. Рисунок 1109), отладка прервется и на экран будет выведено сообщение: «Выполнение макрокоманды прервано пользователем».

В процессе отладки в нижней части окна редактора макрокоманд отображаются следующие области (см. Рисунок 1010):

- **Выполнение...** – окно для вывода сообщений во время отладки, например, командой print;
- **Стек вызовов** – окно стека вызовов;
- **Переменные** – окно вывода значений локальных и глобальных переменных, доступных на текущем шаге выполнения макрокоманды. Если отображаемая переменная представляет из себя таблицу или массив, то при нажатии кнопки ▸, расположенной рядом с именем переменной, доступен просмотр содержимого переменной в развернутом виде.

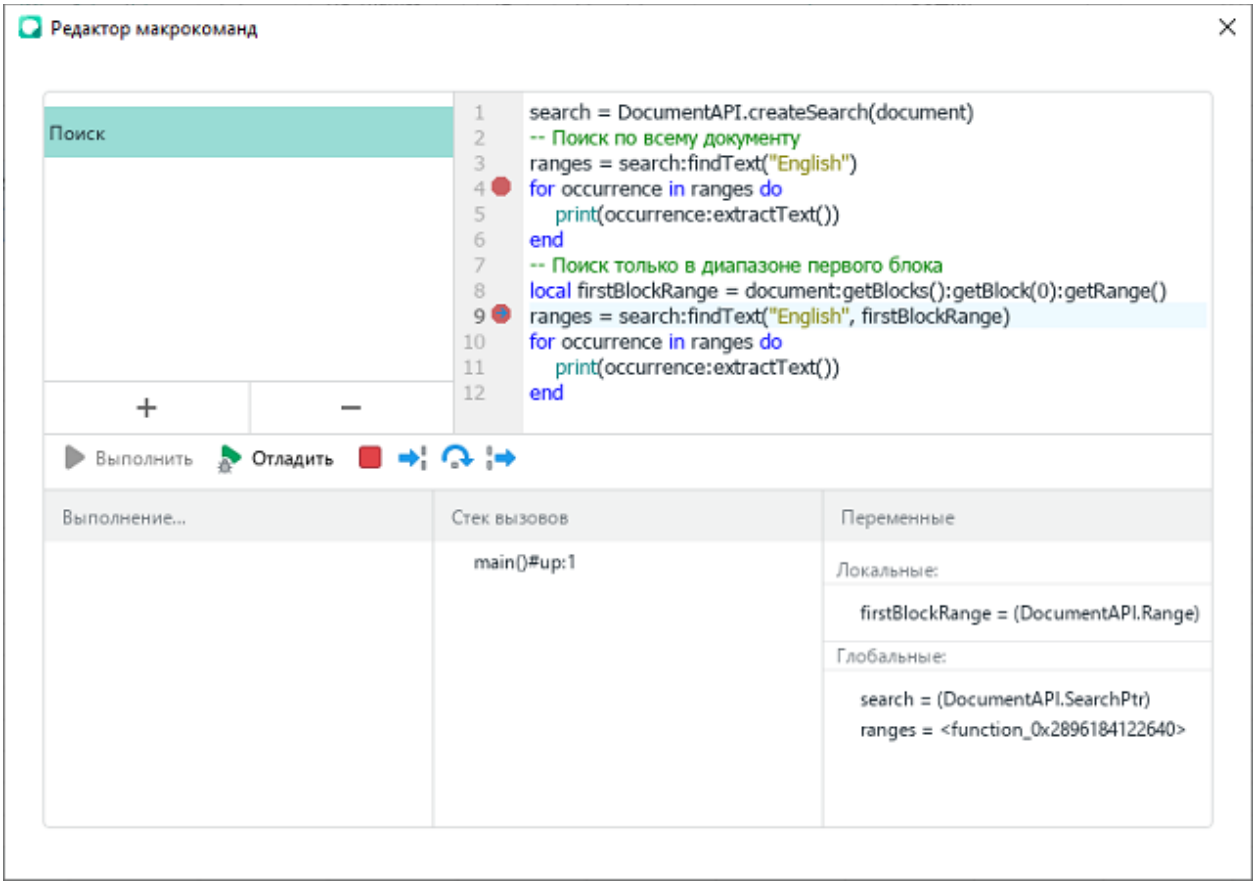


Рисунок 1110 – Окно редактора макрокоманд в процессе отладки макрокоманд

Отладка завершается при достижении конца макрокоманды.

1.2 Пример подготовки и запуска макрокоманды

1.2.1 Редактирование и запуск макрокоманды в текстовом документе

Следующий пример описывает создание и запуск макрокоманды, которая выводит в первой строке текстового документа строку «*HELLO, WORLD!*».

Чтобы создать и запустить макрокоманду, необходимо выполнить следующие действия:

1. Запустить приложение «МойОфис Текст» и открыть редактор макрокоманд. Для этого в командном меню выбрать пункт **Инструменты** > **Макрокоманды** > **Редактор макрокоманд**.
2. Нажать кнопку + для создания новой макрокоманды. Указать имя макрокоманды. По умолчанию новой макрокоманде присваивается имя «Без имени».
3. Ввести текст макрокоманды:

```
range = document:getRange()
startPos = range:getBegin()


textProp = range:getTextProperties()
textProp.italic = true
textProp.allCapitals = true
range:setTextProperties(textProp)
startPos:insertText("Hello, World!")
```

4. Запустить макрокоманду нажатием на кнопку **Выполнить**. В случае успешного завершения работы макрокоманды редактор макрокоманд выведет сообщение «Макрос выполнен успешно».
5. Закрыть окно редактора макрокоманд, чтобы увидеть изменения в текстовом документе. В первой строке документа отобразится текст «*HELLO, WORLD!*».
6. Сохранить текстовый документ. Для этого выбрать в командном меню пункт **Файл** > **Сохранить** / **Файл** > **Сохранить как** или нажать сочетание клавиш **Ctrl**+**S**.

При сохранении необходимо выбрать тип файла «Текстовый документ» одного из форматов: DOCX, XODT, ODT (для электронных таблиц - XLSX, XODS, ODS).

1.2.2 Описание примера работы макрокоманды

Ниже приведено описание макрокоманды, текст которой приведен в разделе Редактирование и запуск макрокоманды в текстовом документе.

С помощью последовательности вызовов устанавливается курсор в начало документа:

```
range = document:getRange()
startPos = range:getBegin()
```

Таблица `DocumentAPI.Document` представляет текущий открытый текстовый документ.

Таблица `DocumentAPI.Range` используется для того, чтобы предоставить доступ к любой части (фрагменту) содержимого документа.

В данном случае, переменная `range` содержит весь документ целиком. Вызов `range:getBegin()` устанавливает курсор в начало фрагмента, а в данном случае – в начало самого документа.

Следующая последовательность вызовов настраивает форматирование для документа:

```
textProp = range:getTextProperties()
textProp.italic = true
textProp.allCapitals = true
range:setTextProperties(textProp)
```

В результате выполнения `range:getTextProperties` переменной `textProp` присваивается экземпляр `TextProperties`, содержащий настройки форматирования текущего фрагмента документа.

Таблица `DocumentAPI.TextProperties` позволяет управлять такими характеристиками как наименование и размер шрифта, цвет, начертание и т.п.

В данном примере устанавливаются две настройки форматирования:

- свойство `textProp.italic` принимает значение **true**, что равносильно нажатию кнопки *K* (**Курсив**) в пользовательском интерфейсе текстового редактора;
- свойство `textProp.allCapitals` принимает значение **true**, что равносильно нажатию кнопки AB (**Все прописные**) в пользовательском интерфейсе текстового редактора.

Следующий вызов `range:setTextProperties(textProp)` применяет новые настройки форматирования для документа. Теперь эти настройки форматирования будут применяться автоматически для вводимого текста.

Последний вызов вставляет в начало документа текст «Hello, World!»:

```
startPos:insertText("Hello, World!")
```

При вставке текста автоматически применяются настройки форматирования, и итоговый текст отображается как «*HELLO, WORLD!*» прописными буквами курсивом.

1.3 Преобразование макрокоманд на языке программирования VBA

Макрокоманды для пакета Microsoft Office, написанные на языке программирования VBA, не предназначены для исполнения в приложениях ПО МойОфис. Макрокоманды на языке программирования VBA предназначены для исполнения только в пакете Microsoft Office под управлением операционной системы семейства Microsoft Windows.

Однако большинство макрокоманд на языке программирования VBA возможно реализовать на языке программирования Lua с использованием объектной модели ПО МойОфис.

ПО МойОфис является кроссплатформенным решением (решением не только для работы в операционных системах семейства Microsoft Windows), поэтому при реализации макрокоманд на основе языка программирования VBA следует принимать во внимание следующие ограничения, связанные с операционными системами семейства Microsoft Windows:

- невозможность обращения к внешним приложениям с помощью технологий Component Object Model (COM);
- невозможность использования внешних динамических библиотек DLL.

Также в настоящее время в редакторе макрокоманд ПО МойОфис существует временное ограничение по работе в тексте макрокоманд с визуальными элементами, такими как выпадающие списки, переключатели и некоторыми другими.

2 ОБЪЕКТНАЯ МОДЕЛЬ МОЙОФИС

МойОфис SDK предоставляет разработчику возможности для управления содержимым текстового и табличного документа. Функции управления сосредоточены в четырех таблицах:

– DocumentAPI – содержит таблицы и функции для представления всех элементов документа, которые поддерживает МойОфис: абзацы, таблицы, рисунки, колонтитулы, операции для работы с текстом, цветом и т.д;
– EditorAPI – содержит функции для управления редакторами МойОфис Текст и МойОфис Таблица;

Вышеописанные таблицы составляют объектную модель МойОфис SDK (см. Рисунок 1111).

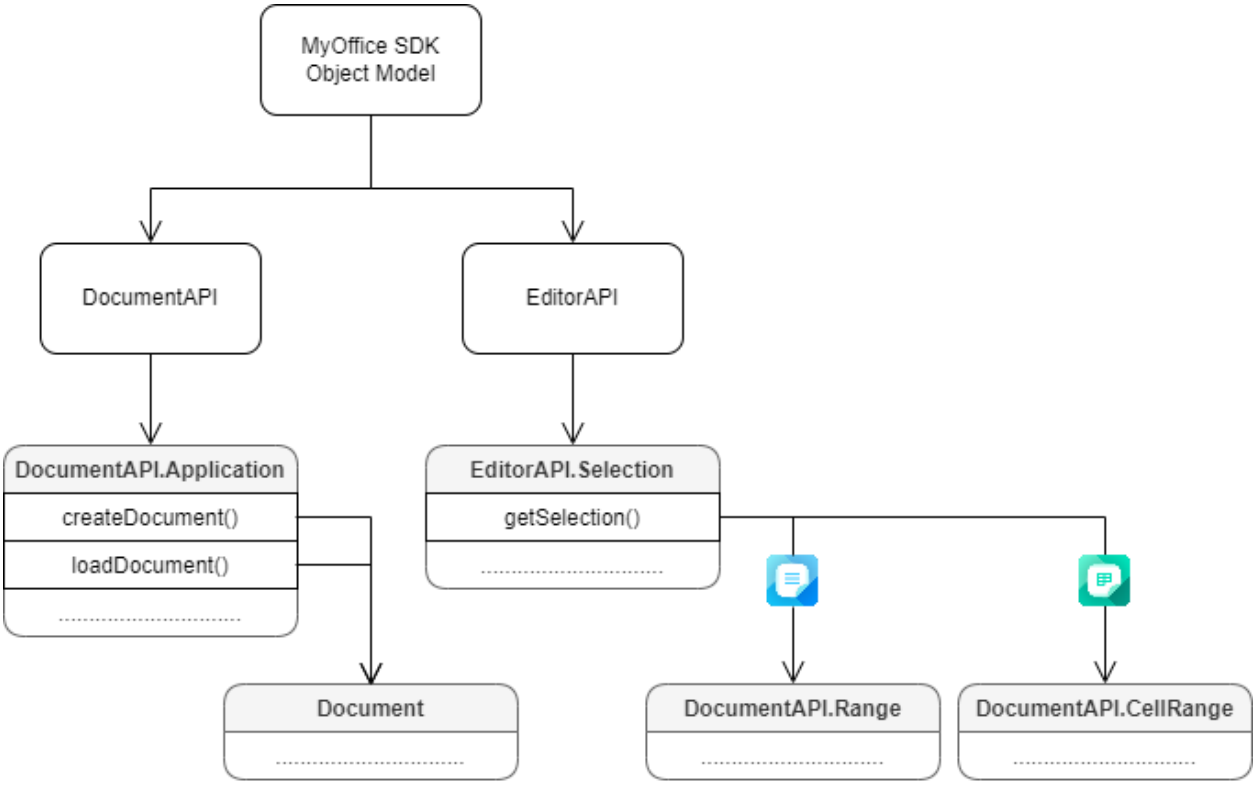


Рисунок 1111 – Объектная модель МойОфис SDK.

3 СПРАВОЧНИК ФУНКЦИЙ DOCUMENTAPI

3.1 Диаграммы

Работа с диаграммами реализована только в табличных документах. На Рисунке 1112 изображена объектная модель таблиц, относящихся к работе с диаграммами.

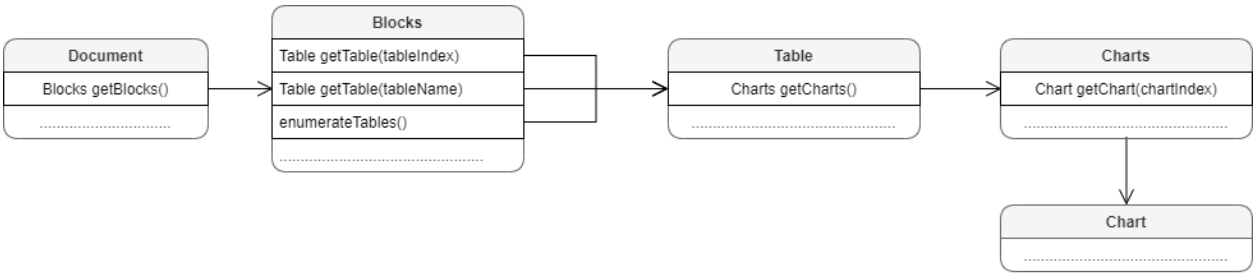


Рисунок 1112 – Объектная модель таблиц для работы с диаграммами

Доступ к списку диаграмм производится через таблицу `DocumentAPI.Table`, соответствующую листу табличного документа.

Пример:

```
local sheetDocumentPage = document:getBlocks():getTable(0)
local charts = sheetDocumentPage:getCharts()
print(charts:getChartsCount())
```

3.1.1 Таблица DocumentAPI.Charts

Таблица `DocumentAPI.Charts` обеспечивает доступ к списку диаграмм (см. Рисунок 1113) табличного документа. Доступ к списку диаграмм осуществляется с помощью метода `Table.getCharts()`.

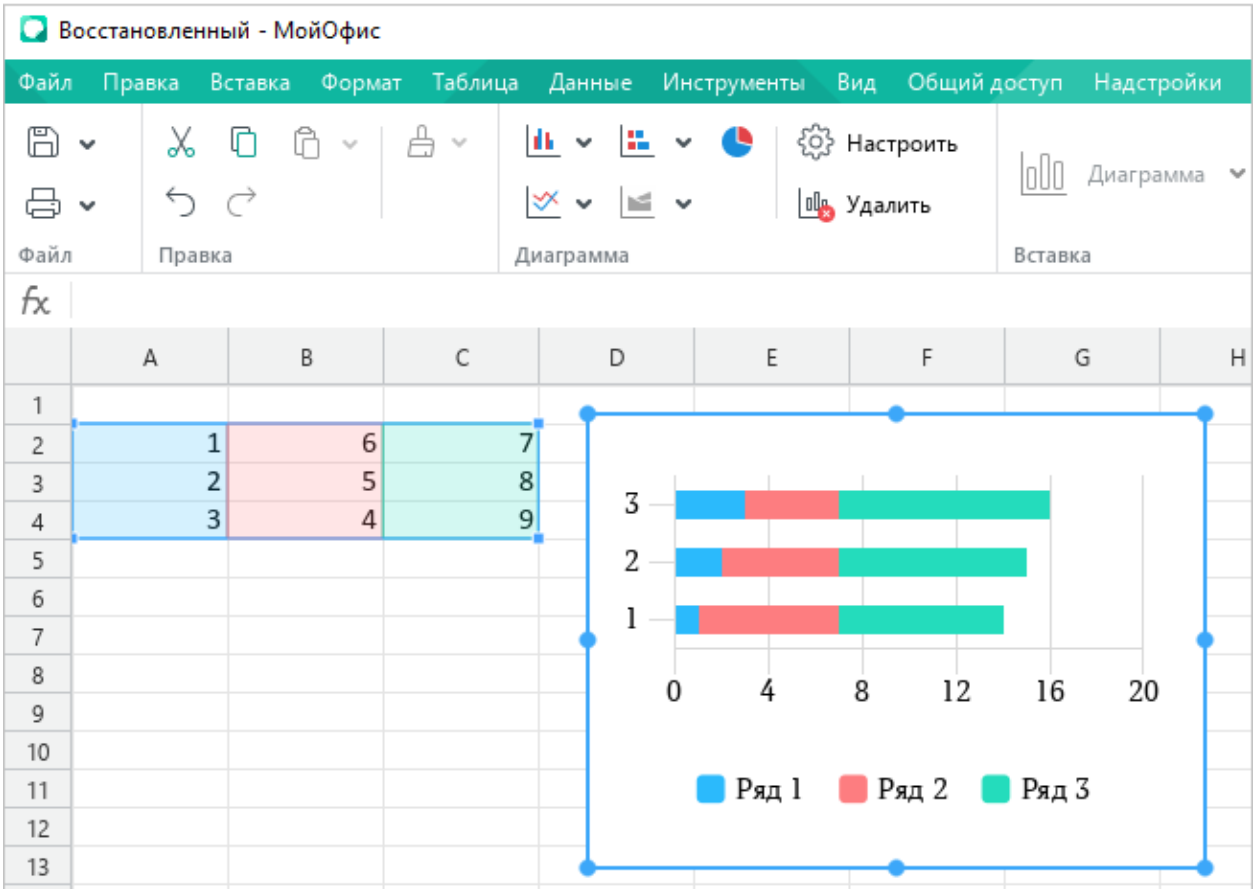


Рисунок 1113 – Пример отображения диаграммы в МойОфис Таблица.

3.1.1.1 Метод Charts:getChartsCount

Метод возвращает общее количество диаграмм в табличном документе.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChartsCount())
```

3.1.1.2 Метод Charts:getChart

Метод возвращает диаграмму `DocumentAPI.Chart` по индексу `chartIndex` в коллекции диаграмм.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRangeAsString())
```

3.1.1.3 Метод Charts:getChartIndexByDrawingIndex

Метод возвращает индекс диаграммы по индексу отрисовки `drawingIndex`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChartIndexByDrawingIndex(0))
```

3.1.2 Таблица DocumentAPI.Chart

Таблица `DocumentAPI.Chart` представляет диаграмму в табличном документе и описывает все ее элементы (заголовок, легенда, тип, данные, диапазон и т.д).

3.1.2.1 Метод Chart:getType

Метод возвращает тип диаграммы `DocumentAPI.ChartType`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getType())
```

3.1.2.2 Метод Chart:setType

Метод устанавливает тип диаграммы `DocumentAPI.ChartType`. Параметр `chartType` - новый тип диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
charts:getChart(0):setType(DocumentAPI.ChartType_LineStacked)
print(charts:getChart(0):getType())
```

3.1.2.3 Метод Chart:getRangesCount

Метод возвращает количество серий диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRangesCount())
```

3.1.2.4 Метод Chart:getRange

Метод возвращает диапазон ячеек `DocumentAPI.ChartRangeInfo` с исходными данными диаграммы. Параметр `rangesIndex` – индекс диапазона.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRange(0).rangeType)
```

3.1.2.5 Метод Chart:getTitle

Метод возвращает заголовок диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getTitle())
```

3.1.2.6 Метод Chart:setRange

Метод задает диапазон `DocumentAPI.CellRangePosition` ячеек с исходными данными для диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
charts:getChart(0):setRange(cellRangePosition)
```

3.1.2.7 Метод Chart:setRect

Метод задает область расположения диаграммы, параметр `rect` – новая область.

Внимание ! Метод устаревший (deprecated), оставлен для обратной совместимости и не рекомендован к использованию.

3.1.2.8 Метод Chart:isEmpty

Метод возвращает `true`, если диаграмма не содержит значений.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):isEmpty())
```

3.1.2.9 Метод Chart:isSolidRange

Метод возвращает `true`, если диапазон исходных данных диаграммы может быть выделен одним прямоугольником и не имеет промежутков.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):isSolidRange())
```

3.1.2.10 Метод Chart:is3D

Метод возвращает `true`, если диаграмма трехмерная.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):is3D())
```

3.1.2.11 Метод Chart:getDirectionType

Метод возвращает направление DocumentAPI.ChartSeriesDirectionType серий диаграммы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getDirectionType())
```

3.1.2.12 Метод Chart:getChartLabels

Метод возвращает коллекцию меток диаграммы типа DocumentAPI.ChartLabelsInfo.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local chart = charts:getChart(0)
local chartLabelsInfo = chart:getChartLabels()
print(chartLabelsInfo.categoriesMode, chartLabelsInfo.seriesNameMode, chartLa
belsInfo.isOneColumnRowChart)
```

3.1.2.13 Метод Chart:getRangeAsString

Метод возвращает диапазон ячеек диаграммы в формате строки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getRangeAsString())
```

3.1.2.14 Метод Chart:applySettings

Метод позволяет обновить параметры текущей выбранной диаграммы.

Вызов:

```
applySettings(cellRange, directionType, title, labelsInfo)
```

Параметры:

– `cellRange` – обновленный диапазон исходных данных диаграммы `DocumentAPI.CellRange`;

– `directionType` – направление серий `DocumentAPI.ChartSeriesDirectionType`;

– `title` – заголовок диаграммы (тип - строка);

– `labelsInfo` – информация о метках диаграммы `DocumentAPI.ChartLabelsInfo`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()


local cellRange = tbl:getCellRange("B3:C4")
local directionType = DocumentAPI.ChartSeriesDirectionType_ByRow
local title = 'Title'
local chartLabelsInfo = DocumentAPI.ChartLabelsInfo(DocumentAPI.ChartLabelsDe
tectionMode_FirstRow, DocumentAPI.ChartLabelsDetectionMode_FirstRow, false)


charts:getChart(0):applySettings(cellRange, directionType, title, chartLabels
Info)
```

3.1.3 Таблица DocumentAPI.ChartLabelsDetectionMode

Таблица `DocumentAPI.ChartLabelsDetectionMode` описывает режимы автоматического определения меток диаграмм. Описание полей таблицы представлено в Таблице 20.

Таблица 20 – Описание полей таблицы `DocumentAPI.ChartLabelsDetectionMode`

Поле	Описание
`DocumentAPI.ChartLabelsDetectionMode_Unknown`	Неопределенный тип.
`DocumentAPI.ChartLabelsDetectionMode_FirstRow`	Метка на первой строке.
`DocumentAPI.ChartLabelsDetectionMode_FirstColumn`	Метка на первой колонке.

Поле	Описание
DocumentAPI.ChartLabelsDetectionMode_NoLabels	Не отрисовывать метки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local chart = charts:getChart(0)
local chartLabels = chart:getChartLabels()
print(chartLabels.categoriesMode, chartLabels.seriesNameMode)
```

3.1.4 Таблица DocumentAPI.ChartLabelsInfo

Таблица DocumentAPI.ChartLabelsInfo описывает настройки автоматического определения меток диаграммы. Инициализируется конструктором, в который передаются параметры:

- categoriesMode - режим автоматического определения меток для категорий, тип DocumentAPI.ChartLabelsDetectionMode;
- seriesNameMode - режим автоматического определения меток для серий, тип DocumentAPI.ChartLabelsDetectionMode;
- oneColumnRow - передается true, если диапазон диаграммы содержит только одну строку или одну колонку.

Описание полей таблицы представлено в Таблице 21.

Таблица 21 – Описание полей таблицы DocumentAPI.ChartLabelsInfo

Поле	Описание	Тип
categoriesMode	Режим автоматического определения меток для категорий	DocumentAPI.ChartLabelsDetectionMode
seriesNameMode	Режим автоматического определения меток для серий	DocumentAPI.ChartLabelsDetectionMode
isOneColumnRowChart	Поле содержит true, если диапазон диаграммы содержит только одну строку или одну колонку	Boolean

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local chart = charts:getChart(0)
```

```
local chartLabelsInfo = chart:getChartLabels()
print(chartLabelsInfo.categoriesMode, chartLabelsInfo.seriesNameMode, chartLabelsInfo.isOneColumnRowChart)
```

3.1.5 Таблица DocumentAPI.ChartRangeInfo

Таблица `DocumentAPI.ChartRangeInfo` описывает серию диаграммы. Инициализируется конструктором, в который передаются следующие параметры:

- `tableRangeInfo` - диапазон ячеек, тип `DocumentAPI.TableRangeInfo`;
- `color` – цвет серии диаграммы, тип `DocumentAPI.ColorRGBA`;
- `hidden` – видимость серии, тип Boolean;
- `rangeType` – тип диапазона исходных данных диаграммы, тип `DocumentAPI.ChartRangeType`.

Описание полей таблицы представлено в Таблице 22.

Таблица 22 – Описание полей таблицы `DocumentAPI.ChartRangeInfo`

Поле	Описание	Тип
`tableRangeInfo`	Исходный диапазон ячеек для серии	`DocumentAPI.TableRangeInfo`
`rangeColor`	Цвет для отрисовки серии	`DocumentAPI.ColorRGBA`
`isHidden`	Задает видимость серии диаграммы	Boolean
`rangeType`	Тип диапазона диаграммы	`DocumentAPI.ChartRangeType`

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)
print(rangeInfo.tableRangeInfo, rangeInfo.rangeColor, rangeInfo.isHidden, rangeInfo.rangeType)
```

3.1.6 Таблица DocumentAPI.ChartRangeType

Таблица `DocumentAPI.ChartRangeType` описывает тип диапазона исходных данных диаграммы. Описание полей таблицы представлено в Таблице 23.

Таблица 23 – Описание полей таблицы `DocumentAPI.ChartRangeType`

Поле	Описание
`DocumentAPI.ChartRangeType_Series`	Серии
`DocumentAPI.ChartRangeType_SeriesName`	Имена серий

Поле	Описание
DocumentAPI.ChartRangeType_Categories	Категории
DocumentAPI.ChartRangeType_DataPoint	Разметка данных

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)


rangeTypes = {"Series", "SeriesName", "Categories", "DataPoint" }
print(rangeTypes[rangeInfo.rangeType + 1])
```

3.1.7 Таблица DocumentAPI.ChartSeriesDirectionType

Таблица DocumentAPI.ChartSeriesDirectionType описывает направление серий диаграмм. Описание полей таблицы представлено в Таблице 24.

Таблица 24 – Описание полей таблицы DocumentAPI.ChartSeriesDirectionType

Поле	Описание
DocumentAPI.ChartSeriesDirectionType_Unknown	Неопределенный тип
DocumentAPI.ChartSeriesDirectionType_ByRow	Серии направлены по строкам
DocumentAPI.ChartSeriesDirectionType_ByColumn	Серии направлены по колонкам

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getDirectionType())
```

3.1.8 Таблица DocumentAPI.ChartType

Таблица DocumentAPI.ChartType описывает все поддерживаемые типы диаграмм. Описание полей таблицы представлено в Таблице 25.

Таблица 25 – Описание полей таблицы DocumentAPI.ChartType

Поле	Описание
DocumentAPI.ChartType_Unknown	Неопределенный тип.
DocumentAPI.ChartType_Bar	Линейчатая диаграмма с группировкой.
DocumentAPI.ChartType_BarStacked	Линейчатая диаграмма с накоплением.
DocumentAPI.ChartType_BarPercentSta	Линейчатая нормированная диаграмма с

Поле	Описание
cked	накоплением.
DocumentAPI.ChartType_Column	Гистограмма с группировкой.
DocumentAPI.ChartType_ColumnStacked	Гистограмма с накоплением.
DocumentAPI.ChartType_ColumnPercentStacked	Нормированная гистограмма с накоплением.
DocumentAPI.ChartType_Line	Стандартный график.
DocumentAPI.ChartType_LineStacked	График с накоплением.
DocumentAPI.ChartType_LinePercentStacked	Нормированный график с накоплением.
DocumentAPI.ChartType_LineWithMarker	Стандартный график с маркерами.
DocumentAPI.ChartType_LineWithMarkerStacked	График с накоплением и маркерами.
DocumentAPI.ChartType_LineWithMarkerPercentStacked	Нормированный график с накоплением и маркерами.
DocumentAPI.ChartType_Area	Стандартная диаграмма с областями.
DocumentAPI.ChartType_AreaStacked	Диаграмма с областями с накоплением.
DocumentAPI.ChartType_AreaPercentStacked	Нормированная диаграмма с областями с накоплением.
DocumentAPI.ChartType_PieAreaPercentStacked	Круговая диаграмма.
DocumentAPI.ChartType_PieExploded	Круговая диаграмма с отделенными секторами.
DocumentAPI.ChartType_Scatter	Диаграмма рассеяния.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
print(charts:getChart(0):getType())
```

3.2 Именованные выражения

Именованное выражение – это выражение (являющееся описанием диапазона или формулой), которому присвоено имя. Преимуществом именованного выражения является его информативность. Именованные выражения упрощают работу с ячейками, также их удобно использовать при работе с формулами. На данный момент доступна возможность работы с именованными выражениями, представляющими собой ссылки на диапазоны

ячеек. Доступ к именованным выражениям осуществляется посредством методов Document:getNamedExpressions() и Table:getNamedExpressions() (см. Рисунок 1114).

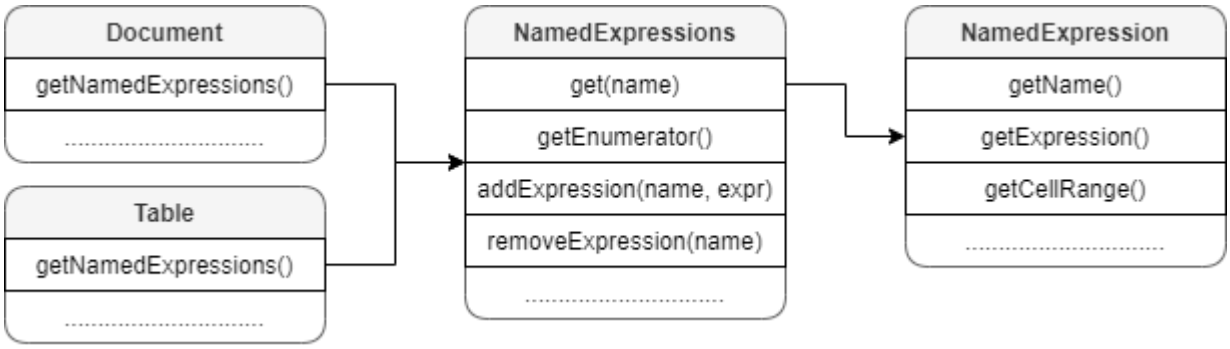


Рисунок 1114 – Таблицы для работы с именованными выражениями

3.2.1 Таблица DocumentAPI.NamedExpressions

Таблица для представления списка именованных выражений. Может быть получена с помощью методов `Document:getNamedExpressions()`, `Table:getNamedExpressions()`.

3.2.1.1 Метод NamedExpressions:get

Возвращает именованное выражение `NamedExpression` по имени `name`, если оно существует.

Пример:

```
local namedExpression = namedExpressions:get("Продажи")
if (namedExpression) then
    print(namedExpression:getName()) -- Продажи
else
    print("No named expression was found")
end
```

3.2.1.2 Метод NamedExpressions:enumerate

Позволяет получить доступ ко всему списку именованных выражений.

Пример:

```
local namedExpressions = sheet:getNamedExpressions()
for namedExpression in namedExpressions:enumerate() do
   print(namedExpression)
end
```

3.2.1.3 Метод NamedExpression:addExpression

Добавляет новое выражение в список именованных выражений, возвращает результат операции NamedExpressionsValidationResult.

Пример:

```
local expressionName = "Покупки"
local expressionValue = "=Формула покупки!$E$6:$E$14"
local validationResult = namedExpressions:addExpression(expressionName, expre
ssionValue)
if (validationResult == DocumentAPI.NamedExpressionsValidationResult_Success)
 then
   print("Named expression was added")
end
```

3.2.1.4 Метод NamedExpressions:removeExpression

Удаляет выражение по заданному имени, возвращает результат операции NamedExpressionsValidationResult.

Пример:

```
local namedExpression = namedExpressions:get(expressionName)
if (namedExpression) then
    local validationResult = namedExpressions:removeExpression(expressionName
)
    if (validationResult == DocumentAPI.NamedExpressionsValidationResult_Succ
ess) then
       print("Named expression was removed")
    end
end
```

3.2.2 Таблица DocumentAPI.NamedExpression

Класс описывает структуру именованного выражения.

Пример:

```
local namedExpressions = sheet:getNamedExpressions()
for namedExpression in namedExpressions:enumerate() do
   print(namedExpression:getName())
   print(namedExpression:getExpression())
   cellRange = namedExpression:getCellRange()
   print(cellRange:getBeginRow(), cellRange:getLastRow())
end
```

3.2.2.1 Метод NamedExpression:getName

Возвращает имя именованного выражения. Пример см. в `DocumentAPI.NamedExpression`.

3.2.2.2 Метод NamedExpression:getExpression

Возвращает текст выражения (формулы). Пример см. в `DocumentAPI.NamedExpression`.

3.2.2.3 Метод NamedExpression:getCellRange

Возвращает диапазон ячеек `DocumentAPI.CellRange`, если выражение является ссылкой на диапазон. Пример см. в `DocumentAPI.NamedExpression`.

3.2.3 Таблица DocumentAPI.NamedExpressionsValidationResult

Таблица `DocumentAPI.NamedExpressionsValidationResult` описывает результат операций `NamedExpressions:addExpression()`, `NamedExpressions:removeExpression()`. Описание полей таблицы представлено в Таблице 26.

Таблица 26 – Описание полей таблицы `DocumentAPI.NamedExpressionsValidationResult`

Поле	Описание
`DocumentAPI.NamedExpressionsValidationResult_Success`	Операция выполнена успешно
`DocumentAPI.NamedExpressionsValidationResult_WrongName`	Неправильный формат имени
`DocumentAPI.NamedExpressionsValidationResult_isUsedInFormula`	Имя уже используется в формуле

3.3 Макрокоманды

Таблица `DocumentAPI.Scripts` предоставляет доступ к списку макрокоманд документа. На Рисунке 1115 изображена объектная модель таблиц, относящихся к работе с макрокомандами.

Таблица `DocumentAPI.Scripts` предназначена для доступа к списку макрокоманд, доступна через метод `document:getScripts()`, таблица `DocumentAPI.Scripting` служит для запуска макрокоманд, доступна через `DocumentAPI.createScripting(document)`.

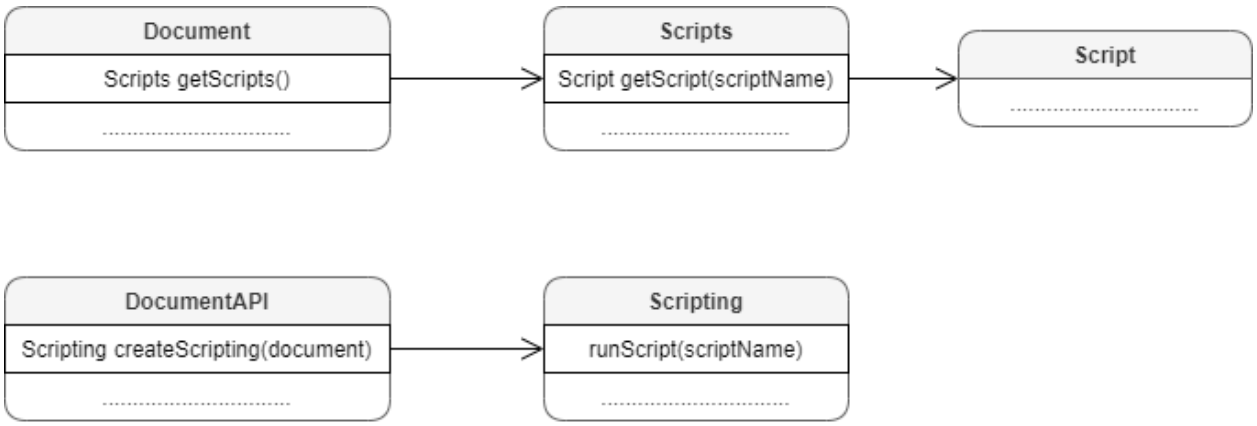


Рисунок 1115 – Объектная модель таблиц для работы с макрокомандами

3.3.1 Таблица DocumentAPI.Scripts

Таблица `DocumentAPI.Scripts` предоставляет доступ к списку макрокоманд документа. Коллекцию макрокоманд `DocumentAPI.Scripts` можно получить из документа посредством вызова метода `document:getScripts()`.

Пример:

```
local scripts = document:getScripts()
for script in scripts:enumerate() do
   print(script:getName())
   print(script:getBody())
end
```

3.3.1.1 Метод Scripts:getScript

Метод возвращает таблицу `DocumentAPI.Script`, описывающую макрокоманду. В качестве аргумента используется имя макрокоманды.

Пример:

```
local scripts = document:getScripts()
local script = scripts:getScript("Enumerate scripts for document")
print(script:getName())
```

3.3.1.2 Метод Scripts:setScript

Метод добавляет макрокоманду в текущий документ. Если макрокоманда с таким именем уже существует, будет обновлено ее содержимое.

Пример:

```
local scripts = document:getScripts()
local script_name = "Enumerate scripts for document"
```

```
local script_code = "local scripts = document:getScripts()\nfor script in scr
ipts:enumerate() do\nprint(script:getName())\nend"
scripts:setScript(script_name, script_code)
```

3.3.1.3 Метод Scripts:removeScript

Метод удаляет макрокоманду из текущего документа. В качестве аргумента используется имя макрокоманды.

Пример:

```
local scripts = document:getScripts()
scripts:removeScript("Enumerate scripts for document")
```

3.3.1.4 Метод Scripts:enumerate

Метод возвращает коллекцию макрокоманд для их дальнейшего перечисления.

Пример:

```
for script in document:getScripts():enumerate() do
    print(script:getName())
end
```

3.3.2 Таблица DocumentAPI.Script

Таблица `DocumentAPI.Script` предназначена для управления отдельной макрокомандой. Таблица содержит поля `Name` и `Body`.

3.3.2.1 Таблица DocumentAPI.Scripting

Таблица `DocumentAPI.Scripting` может быть получена путем вызова `DocumentAPI.createScripting()` и содержит метод `runScript`, который используется для запуска макрокоманды.

3.3.2.1.1 Метод Scripting:runScript

Метод предназначен для запуска макрокоманды, хранящейся в документе. В качестве аргумента передается имя макрокоманды.

Пример:

```
scripting = DocumentAPI.createScripting(document)
scripting:runScript("Enumerate scripts for document")
```

3.3.2.2 Метод Script:getName

Метод возвращает имя макрокоманды.

Пример:

```
local scripts = document:getScripts()
local sc = scripts:getScript("Enumerate scripts for document")
print(sc:getName())
```

3.3.2.3 Метод Script:setName

Метод устанавливает имя для макрокоманды.

Пример:

```
local scripts = document:getScripts()
local sc = scripts:getScript("Enumerate scripts for document")
sc:setName("Enumerate scripts for current document")
```

3.3.2.4 Метод Script:getBody

Метод возвращает текст макрокоманды в виде строки.

Пример:

```
local scripts = document:getScripts()
local script = scripts:getScript("Enumerate scripts for document")
local scriptBody = script:getBody()
print(scriptBody)
```

3.3.2.5 Метод Script:setBody

Метод устанавливает текст макрокоманды, полностью заменяя уже имеющийся текст.

Пример:

```
local scripts = document:getScripts()
local script = scripts:getScript("Enumerate scripts for document")
script:setBody("local scripts = document:getScripts()\nfor script in
scripts:enumerate() do\nprint(script:getName())\nend")
```

3.3.3 Функция DocumentAPI.createScripting

Функция `DocumentAPI.createScripting` возвращает таблицу `DocumentAPI.Scripting`. В качестве параметра используется текущий документ.

Пример:

```
scripting = DocumentAPI.createScripting(document)
```

3.4 Разделы (секции) документа

На Рисунке 1116 изображена объектная модель таблиц, относящихся к работе с секциями текстового документа.

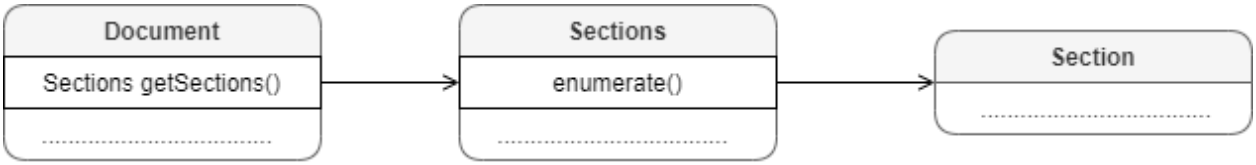


Рисунок 1116 – Объектная модель таблиц для работы с секциями

Секция в текстовом документе - это раздел, который содержит страницы с одинаковыми параметрами, а также одинаковыми верхними и нижними колонтитулами.

Доступ к секциям текстового документа может быть осуществлен одним из следующих способов:

- получение таблицы `DocumentAPI.Sections` с помощью вызова `document:getSections()`;
- перечисление всех доступных секций `DocumentAPI.Section` с помощью вызова `document:enumerateSections()`;
- получение секции `DocumentAPI.Section` вызовом метода `Block.getSection()` для блока, который входит в секцию.

Примеры:

```
local sections = document:getSections()
for section in sections:enumerate() do
    local properties = section:getPageProperties()
    print(properties.width)
    print(properties.height)
end

local sections = document:enumerateSections()
for section in sections do
  print(section:getPageProperties().width)
end

local section = document:getBlocks():getBlock(0):getSection()
local properties = section:getPageProperties()
```

3.4.1 Таблица DocumentAPI.Sections

Таблица `DocumentAPI.Sections` представляет интерфейс для доступа к коллекции секций документа. Описание секции см. в разделе `DocumentAPI.Section`.

3.4.1.1 Метод Sections:enumerate

Метод возвращает коллекцию секций документа.

Пример:

```
local sections = document:getSections()
for section in sections:enumerate() do
  local properties = section:getPageProperties()
  print(properties.width)
  print(properties.height)
end
```

3.4.2 Таблица DocumentAPI.Section

Таблица `DocumentAPI.Section` представляет собой раздел в документе.

3.4.2.1 Метод Section:setPageProperties

Метод устанавливает параметры `DocumentAPI.PageProperties` страниц, находящихся в разделе.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local properties = section:getPageProperties()
properties.width = 100
properties.height = 200
properties.margins.left = 10
section:setPageProperties(properties)
```

3.4.2.2 Метод Section:getPageProperties

Метод возвращает параметры страниц раздела `DocumentAPI.PageProperties`.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local properties = section:getPageProperties()
print(properties.width)
print(properties.height)
```

```
print(properties.margins.left)
print(properties.margins.top)
```

3.4.2.3 Метод Section:setPageOrientation

Метод задает ориентацию страниц раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
section:setPageOrientation(DocumentAPI.PageOrientation_Landscape)
local orientation = section:getPageOrientation()
print(orientation)
```

3.4.2.4 Метод Section:getPageOrientation

Метод возвращает ориентацию страниц раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local orientation = section:getPageOrientation()
print(orientation)
```

3.4.2.5 Метод Section:getRange

Метод возвращает диапазон DocumentAPI.Range в документе, соответствующий данному разделу.

Пример:

```
local sections = document:enumerateSections()
for section in sections do
  print(section:getRange():extractText())
end
```

3.4.2.6 Метод Section:getHeaders

Метод возвращает коллекцию DocumentAPI.HeadersFooters верхних колонтитулов данного раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    if (header:getType() == DocumentAPI.HeaderFooterType_Header) then
       print("Header") else print("Footer")
```

```
    end
end
```

3.4.2.7 Метод Section:getFooters

Метод возвращает коллекцию DocumentAPI.HeadersFooters нижних колонтитулов данного раздела.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local footers = section:getFooters()
for footer in footers:enumerate() do
    if (footer:getType() == DocumentAPI.HeaderFooterType_Header) then
       print("Header") else print("Footer")
    end
end
```

3.4.3 Таблица DocumentAPI.HeadersFooters

Таблица DocumentAPI.HeadersFooters представляет коллекцию верхних и нижних колонтитулов раздела (см. Рисунок 1117). Доступ к колонтитулам осуществляется посредством методов Section.getHeaders(), Section.getFooters().

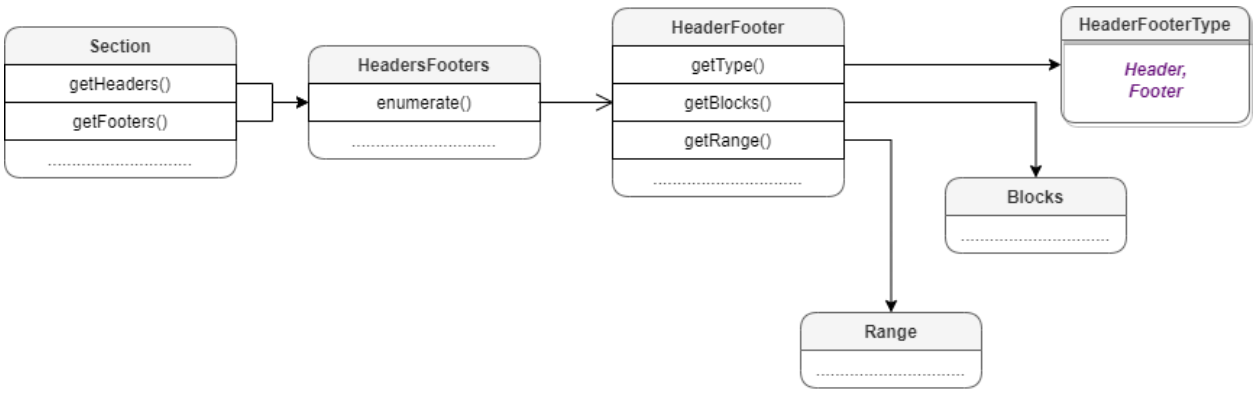


Рисунок 1117 – Таблицы колонтитулов

3.4.3.1 Метод HeadersFooters:enumerate

Метод возвращает коллекцию колонтитулов.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
```

```
    if (header:getType() == DocumentAPI.HeaderFooterType_Header) then
       print("Header") else print("Footer")
    end
end
```

3.4.4 Таблица DocumentAPI.HeaderFooter

Таблица `DocumentAPI.HeaderFooter` определяет колонтитул текстового документа.

3.4.4.1 Метод HeaderFooter:getType

Метод предоставляет информацию о типе колонтитула (`DocumentAPI.HeaderFooterType`).

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    if (header:getType() == DocumentAPI.HeaderFooterType_Header) then
       print("Header") else print("Footer")
    end
end
```

3.4.4.2 Метод HeaderFooter:getBlocks

Метод предоставляет доступ к блокам (`DocumentAPI.Blocks`), которые содержатся в колонтитуле.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    for block in header:getBlocks():enumerate() do
        print(block:getRange():extractText())
    end
end
```

3.4.4.3 Метод HeaderFooter:getRange

Метод предоставляет диапазон (`DocumentAPI.Range`) с содержанием верхнего или нижнего колонтитулов.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local headers = section:getHeaders()
for header in headers:enumerate() do
    print(header:getRange():extractText())
end
```

3.4.5 Таблица DocumentAPI.HeaderFooterType

Типы колонтитулов представлены в Таблице 27.

Таблица 27 – Типы колонтитулов

Наименование константы	Описание
DocumentAPI.HeaderFooterType_Header	Верхний колонтитул
DocumentAPI.HeaderFooterType_Footer	Нижний колонтитул

3.4.6 Таблица DocumentAPI.PageOrientation

Типы ориентации страницы представлены в Таблице 28. Данная константа может быть использована для получения / установки ориентации страниц для секции или документа.

Таблица 28 – Типы ориентации страницы

Наименование константы	Описание
DocumentAPI.PageOrientation_Landscape	Альбомная ориентация страницы
DocumentAPI.PageOrientation_Portrait	Портретная ориентация страницы

Примеры:

```
local section = document:getBlocks():getBlock(0):getSection()
section:setPageOrientation(DocumentAPI.PageOrientation_Landscape)
print(section:getPageOrientation())

local section = document:setPageOrientation(DocumentAPI.PageOrientation_Portrait)
local section = document:getBlocks():getBlock(0):getSection()
print(section:getPageOrientation())
```

3.4.7 Таблица DocumentAPI.Insets

Таблица DocumentAPI.Insets предназначина для задания полей, например, страницы. DocumentAPI.Insets представлено в Таблице 29. Используется в поле margins таблицы DocumentAPI.PageProperties.

Таблица 29 – Описание полей таблицы `DocumentAPI.Insets`

Поле	Тип	Описание
`DocumentAPI.Insets.left`	number	Левая граница поля.
`DocumentAPI.Insets.top`	number	Верхняя граница поля.
`DocumentAPI.Insets.right`	number	Правая граница поля.
`DocumentAPI.Insets.bottom`	number	Нижняя граница поля.

Пример:

```
local insets = DocumentAPI.Insets()
insets.left = 10.0
print(insets.left)
```

3.4.8 Таблица DocumentAPI.PageProperties

Таблица `DocumentAPI.PageProperties` предоставляет такие свойства страницы как высота, ширина, размеры полей. Описание полей приведено в Таблице 30. Используется в `Document.setPageProperties()`, `Section.getPageProperties()`, `Section.setPageProperties()`.

Таблица 30 – Описание полей таблицы `DocumentAPI.PageProperties`

Поле	Описание
`DocumentAPI.PageProperties.height`	Высота страницы
`DocumentAPI.PageProperties.width`	Ширина страницы
`DocumentAPI.PageProperties.margins`	Поля страницы, тип - `Insets`

Примеры:

```
local section = document:getBlocks():getBlock(0):getSection()
local pageProperties = section:getPageProperties()
pageProperties.width = 100
pageProperties.height = 200
pageProperties.margins.left = 10
section:setPageProperties(pageProperties)

local pageProperties = DocumentAPI.PageProperties()
pageProperties.width = 100
pageProperties.height = 200
document:setPageProperties(pageProperties)

local pageProperties = DocumentAPI.PageProperties(100, 200)
document:setPageProperties(pageProperties)
```

3.4.8.1 Метод PageProperties:__eq

Метод позволяет использовать оператор сравнения == для определения эквивалентности содержимого двух структур DocumentAPI.PageProperies.

3.4.8.2 Метод PageProperties:__ne

Метод позволяет использовать оператор сравнения != для определения неэквивалентности содержимого двух структур DocumentAPI.PageProperies.

3.5 Блоки, параграфы

3.5.1 Таблица DocumentAPI.Blocks

Таблица DocumentAPI.Blocks обеспечивает доступ к блокам DocumentAPI.Block документа или диапазона документа (см. Рисунок 1118). Таблица DocumentAPI.Blocks может быть получена вызовом метода Document:getBlocks или HeaderFooter:getBlocks.

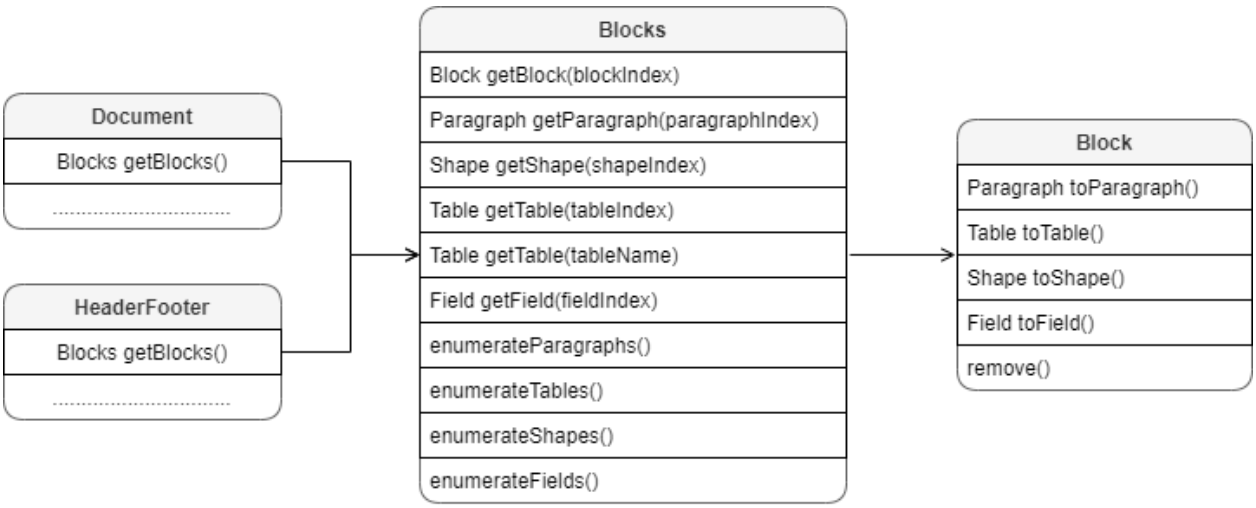


Рисунок 1118 – Объектная модель таблицы DocumentAPI.Blocks

3.5.1.1 Метод Blocks:getBlock

Возвращает объект типа DocumentAPI.Block по заданному индексу. Нумерация индексов начинается с нуля.

Пример:

```
local block = document:getBlocks():getBlock(0)
```

3.5.1.2 Метод Blocks:getParagraph

Возвращает абзац с указанным индексом. Нумерация индексов начинается с нуля.

Пример:

```
local para = document:getBlocks():getParagraph(0)
```

3.5.1.3 Метод Blocks:getTable

Для табличного документа возвращает лист (`worksheet`), для текстового документа возвращает таблицу. Параметры поиска - индекс или имя таблицы. Нумерация листов начинается с нуля.

Пример:

```
local table = document:getBlocks():getTable(0)
```

В качестве параметра метода также можно указать имя таблицы.

Пример:

```
local table = document:getBlocks():getTable("Sheet1")
```

3.5.1.4 Метод Blocks:getShape

Возвращает фигуру `DocumentAPI.Shape` по заданному индексу.

Пример:

```
local shape = document:getBlocks():getShape(0)
```

3.5.1.5 Метод Blocks:getField

Возвращает объект типа `DocumentAPI.Field` по заданному индексу.

Пример:

```
local field = document:getBlocks():getField(0)
```

3.5.1.6 Метод Blocks:enumerate

Позволяет перечислить объекты типа `DocumentAPI.Block`.

Пример:

```
for block in document:getBlocks():enumerate() do
    print(block:getRange():extractText())
end
```

3.5.1.7 Метод Blocks:enumerateParagraphs

Позволяет реализовать перечисление абзацев DocumentAPI.Paragraph.

Пример:

```
for paragraph in document:getBlocks():enumerateParagraphs() do
    print(paragraph:getRange():extractText())
end
```

3.5.1.8 Метод Blocks:enumerateTables

Позволяет перечислить объекты типа DocumentAPI.Table.

Пример:

```
for table in document:getBlocks():enumerateTables() do
    print(table:getName())
end
```

3.5.1.9 Метод Blocks:enumerateShapes

Позволяет перечислить объекты типа DocumentAPI.Shape.

Пример:

```
for shape in document:getBlocks():enumerateShapes() do
    print(shape:getShapeProperties())
end
```

3.5.1.10 Метод Blocks:enumerateFields

Позволяет перечислить объекты типа DocumentAPI.Field.

Пример:

```
for field in document:getBlocks():enumerateFields() do
    print(field:getRange():extractText())
end
```

3.5.2 Таблица DocumentAPI.Block

Таблица DocumentAPI.Block является базовой для всех блоков документа. От нее наследуются таблицы Paragraph, Table, Shape, Field (см. Рисунок 1119).

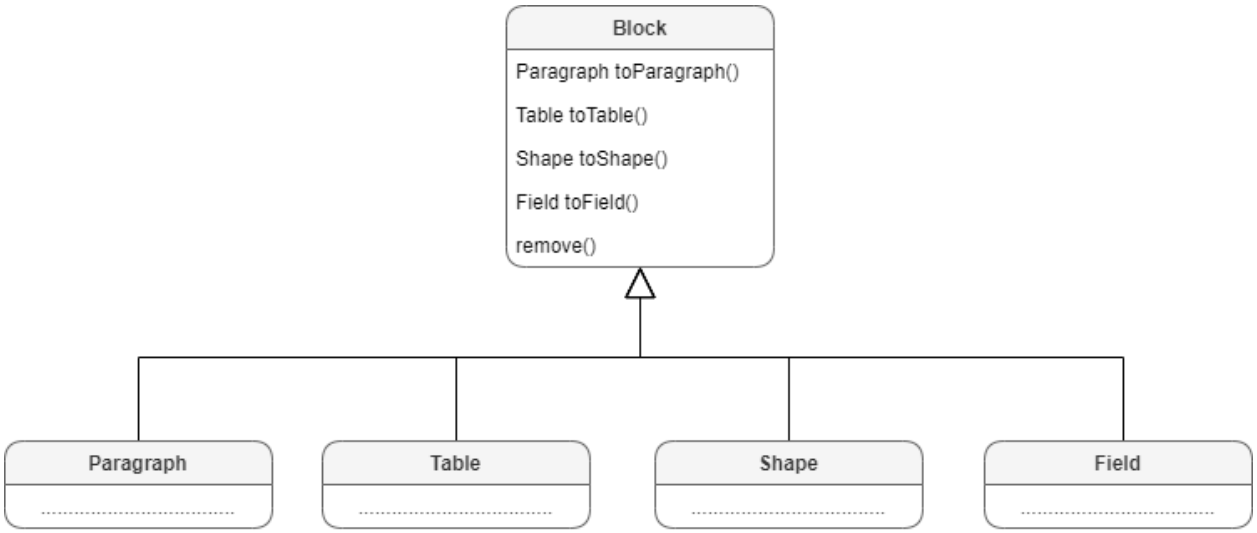


Рисунок 1119 – Объектная модель таблицы `DocumentAPI.Block`

3.5.2.1 Методы toParagraph, toTable, toShape, toField

Преобразует объект `DocumentAPI.Block` в объект соответствующего типа.

Пример:

```
local paragraph = document:getBlocks():getBlock(0):toParagraph()
local para_props = paragraph:getParagraphProperties()
```

3.5.2.2 Метод Block.getRange

Возвращает диапазон `DocumentAPI.Range`, в котором содержится данный блок.

Пример:

```
local range = document:getBlocks():getBlock(0):getRange()
print(range:extractText())
```

3.5.2.3 Метод Block.remove

Удаляет блок из документа. Текущий экземпляр объекта `DocumentAPI.Block` становится недействительным.

Пример:

```
document:getBlocks():getBlock(0):remove()
```

3.5.2.4 Метод Block.getSection

Метод возвращает раздел `DocumentAPI.Section`, содержащий блок.

Пример:

```
local section = document:getBlocks():getBlock(0):getSection()
local pageProperties = section:getPageProperties()
```

3.5.3 Таблица DocumentAPI.Field

Таблица `Field` предназначена для реализации некоторых полей, например, содержания.

3.5.4 Таблица DocumentAPI.Paragraphs

Таблица `DocumentAPI.Paragraphs` предоставляет доступ к коллекции абзацев типа `DocumentAPI.Paragraph` (см. Рисунок 1120). Коллекция абзацев может быть получена из таблицы `DocumentAPI.Range` посредством использования вызова `Range:getParagraphs()`.

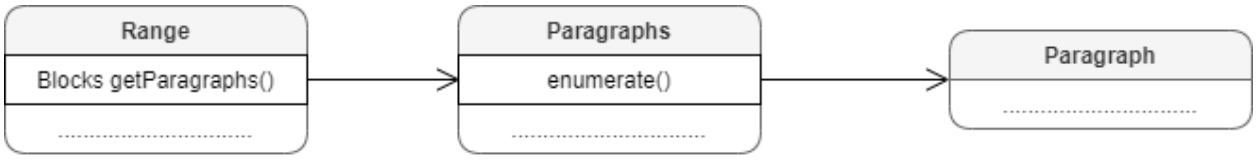


Рисунок 1120 – Объектная модель для работы со списком абзацев

Пример для текстового документа:

```
local paragraphs = document:getRange():getParagraphs()
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local paragraphs = cell:getRange():getParagraphs()
```

3.5.4.1 Метод Paragraphs:setListSchema

Метод устанавливает тип маркированного или нумерованного списка `DocumentAPI.ListSchema`. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
```

3.5.4.2 Метод Paragraphs:setListLevel

Метод устанавливает глубину вложенности элемента списка. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
paragraphs:setListLevel(1)
```

3.5.4.3 Метод Paragraphs:increaseListLevel

Метод увеличивает уровень списка на единицу. В случае, если максимальный уровень уже установлен, увеличения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
paragraphs:increaseListLevel()
```

3.5.4.4 Метод Paragraphs:decreaseListLevel

Метод уменьшает уровень списка на единицу. В случае, если минимальный уровень уже установлен, уменьшения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraphs = document:getRange():getParagraphs()
paragraphs:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
paragraphs:decreaseListLevel()
```

3.5.4.5 Метод Paragraphs:enumerate

Метод позволяет перечислить коллекцию абзацев.

Пример для текстового документа:

```
local paragraphs = document:getRange():getParagraphs()
for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    print(para_props.alignment)
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()


for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    print(para_props.alignment)
end
```

3.5.5 Таблица DocumentAPI.Paragraph

Таблица `DocumentAPI.Paragraph` предоставляет доступ к свойствам абзаца (см. Рисунок 1121).

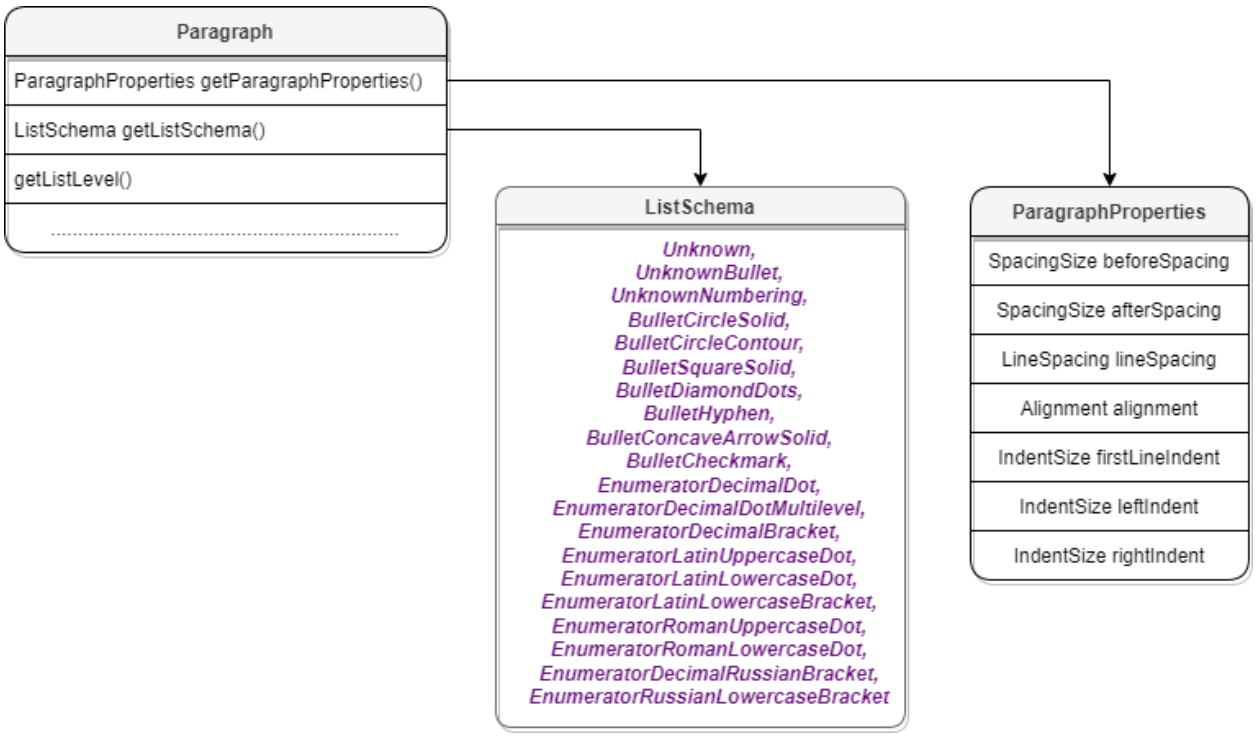


Рисунок 1121 – Объектная модель таблиц для работы со свойствами параграфа

3.5.5.1 Метод Paragraph:getParagraphProperties

Метод предоставляет доступ к таблице свойств форматирования абзаца `DocumentAPI.ParagraphProperties`, таким как выравнивание текста, межстрочные интервалы, отступы и т. д.

Пример для текстового документа:

```
local para = document:getBlocks():getParagraph(0)
local para_props = para:getParagraphProperties()
print(para_props.afterSpacing)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    print(para_props.afterSpacing)
end
```

3.5.5.2 Метод Paragraph:setParagraphProperties

Метод предназначен для обновления таблицы свойств форматирования абзаца DocumentAPI.ParagraphProperties.

Пример для текстового документа:

```
local para = document:getBlocks():getParagraph(0)
local para_props = para:getParagraphProperties()
para_props.alignment = DocumentAPI.Alignment_Right
para:setParagraphProperties(para_props)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    local para_props = para:getParagraphProperties()
    para_props.alignment = DocumentAPI.Alignment_Right
    para:setParagraphProperties(para_props)
end
```

3.5.5.3 Метод Paragraph:getListSchema

Метод возвращает схему форматирования абзаца DocumentAPI.ListSchema либо значение nil, если схема нумерации не установлена для абзаца. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
local schema = paragraph:getListSchema()
```

3.5.5.4 Метод Paragraph:setListSchema

Метод позволяет установить тип маркированного или нумерованного списка DocumentAPI.ListSchema. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
```

3.5.5.5 Метод Paragraph:getListLevel

Метод позволяет получить глубину вложенности элемента списка. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
local level = paragraph:getListLevel()
```

3.5.5.6 Метод Paragraph:setListLevel

Метод позволяет установить глубину вложенности элемента списка.

Значение может быть равным nil, если схема нумерации не установлена для абзаца. В этом случае будет установлено минимальное значение. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
local level = paragraph:setListLevel(1)
```

3.5.5.7 Метод Paragraph:increaseListLevel

Метод позволяет увеличить на единицу глубину вложенности элемента списка. В случае, если максимальный уровень уже установлен, увеличения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
paragraph:increaseListLevel()
```

3.5.5.8 Метод Paragraph:decreaseListLevel

Метод позволяет уменьшить на единицу глубину вложенности элемента списка. В случае, если минимальный уровень уже установлен, уменьшения не происходит. Данный метод используется только в текстовом документе.

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
paragraph:decreaseListLevel()
```

3.5.6 Таблица DocumentAPI.ListSchema

Типы схем форматирования списков, которые могут быть применены к абзацам текста представлены в Таблице 31. Данные константы используются в методах Paragraph:getListSchema(), Paragraph:setListSchema().

Таблица 31 – Типы схем форматирования списков

Наименование константы	Описание
DocumentAPI.ListSchema_Unknown	Схема не определена
DocumentAPI.ListSchema_UnknownBullet	Список без маркера
DocumentAPI.ListSchema_UnknownNumbering	Нумерация без номера
DocumentAPI.ListSchema_BulletCircleSolid	Список с маркерами в виде заполненного круга
DocumentAPI.ListSchema_BulletCircleContour	Список с маркерами в виде окружности
DocumentAPI.ListSchema_BulletSquareSolid	Список с маркерами в виде квадрата
DocumentAPI.ListSchema_BulletDiamondDots	Список с маркерами в виде четырех ромбов
DocumentAPI.ListSchema_BulletHyphen	Список с маркерами в виде

Наименование константы	Описание
	дефиса
DocumentAPI.ListSchema_BulletConcaveArrowSolid	Список с маркерами в виде вогнутой стрелки.
DocumentAPI.ListSchema_BulletCheckmark	Список с маркерами в виде галочки.
DocumentAPI.ListSchema_EnumeratorDecimalDot	Десятичная нумерация с точкой.
DocumentAPI.ListSchema_EnumeratorDecimalDotMultiLevel	Многоуровневая десятичная нумерация с точкой
DocumentAPI.ListSchema_EnumeratorDecimalBracket	Десятичная нумерация со скобкой
DocumentAPI.ListSchema_EnumeratorLatinUppercaseDot	Нумерация латинскими прописными буквами с точкой
DocumentAPI.ListSchema_EnumeratorLatinLowercaseDot	Нумерация латинскими строчными буквами с точкой
DocumentAPI.ListSchema_EnumeratorLatinLowercaseBracket	Нумерация латинскими строчными буквами со скобкой
DocumentAPI.ListSchema_EnumeratorRomanUppercaseDot	Нумерация римскими прописными цифрами с точкой
DocumentAPI.ListSchema_EnumeratorRomanLowercaseDot	Нумерация римскими строчными цифрами с точкой
DocumentAPI.ListSchema_EnumeratorDecimalRussianBracket	Десятичная нумерация через запятую со скобкой
DocumentAPI.ListSchema_EnumeratorRussianLowercaseBracket	Нумерация с русскими строчными буквами со скобкой

Пример:

```
local paragraph = document:getBlocks():getParagraph(0)
paragraph:setListSchema(DocumentAPI.ListSchema_BulletCircleSolid)
```

3.5.7 Таблица DocumentAPI.ParagraphProperties

Таблица `DocumentAPI.ParagraphProperties` предназначена для управления свойствами форматирования (см. Рисунок 1122). Таблица `DocumentAPI.ParagraphProperties` используется в методах `Paragraph:getParagraphProperties` и `Paragraph:setParagraphProperties`.

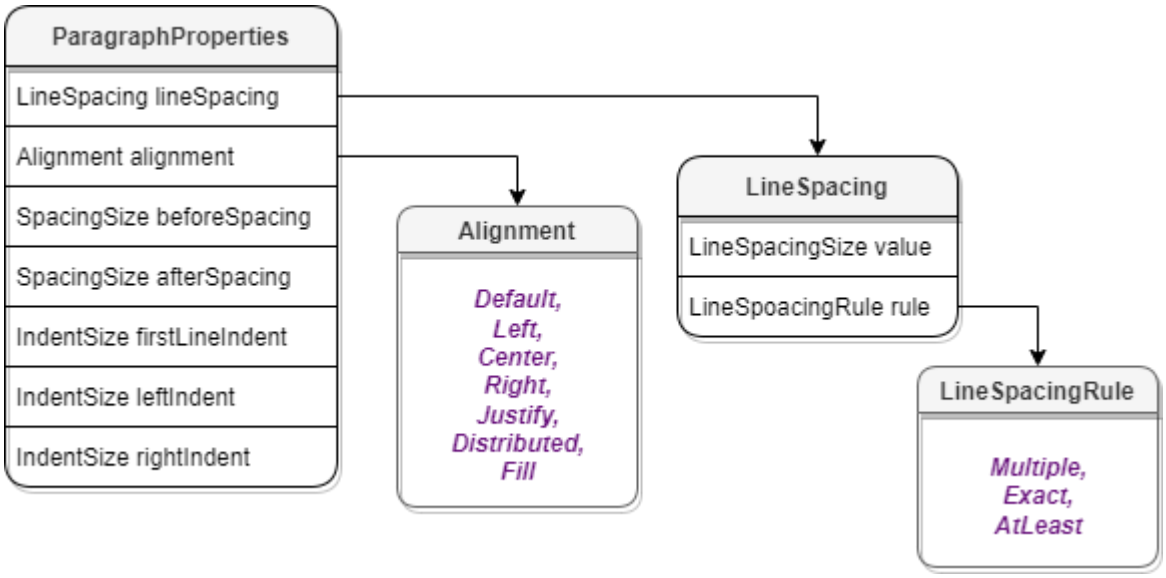
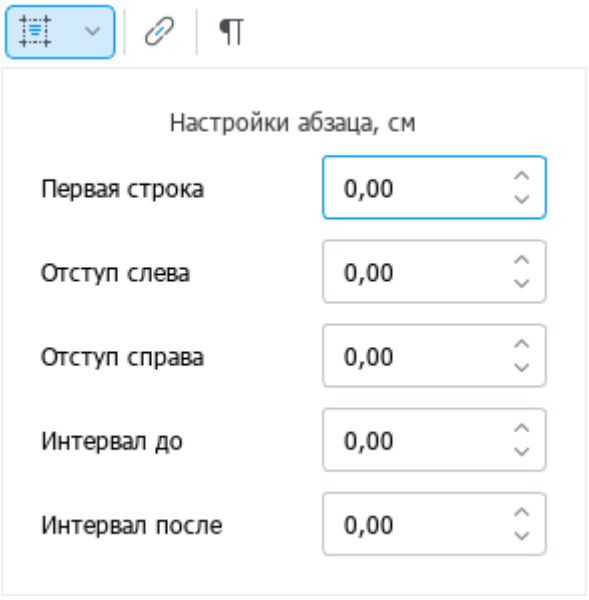


Рисунок 1122 – Объектная модель таблиц для работы со свойствами параграфа

Описание полей таблицы `DocumentAPI.ParagraphProperties` представлено в Таблице 32.

Таблица 32 – Описание полей таблицы `DocumentAPI.ParagraphProperties`

Поле	Описание
`ParagraphProperties.beforeSpacing`	Установка величины расстояния до абзаца. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца,** (см. рисунок выше), в поле **Интервал до** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).
`ParagraphProperties.afterSpacing`	Установка величины расстояния после абзаца. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца**, в поле **Интервал после** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).

Поле	Описание
	
`ParagraphProperties.lineSpacing`	Расстояние между строк одного абзаца (межстрочный интервал), `LineSpacing`.
`ParagraphProperties.alignment`	Выравнивание текстового фрагмента по горизонтали. Список допустимых значений находится в разделе `Alignment`.
`ParagraphProperties.firstLineIndent`	Расстояние от левого поля документа до первой строки в абзаце с учетом отступа слева. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца**, (см. рисунок выше), в поле **Первая строка** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).
`ParagraphProperties.leftIndent`	Расстояние от левого поля документа до абзаца (отступ слева). При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца**, (см. рисунок выше), в поле **Отступ слева** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).
`ParagraphProperties.rightIndent`	Расстояние от правого поля документа до абзаца. При работе с пользовательским интерфейсом приложения соответствует значению, указанному в диалоговом окне **Настройки абзаца**, (см. рисунок выше), в поле **Отступ справа** (подробнее см. в документе «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).

Пример для текстового документа:

```
local para = document:getBlocks():getParagraph(0)
local para_props = para:getParagraphProperties()
--
para_props.afterSpacing = 28.3 -- значение соответствует 1 см
para_props.beforeSpacing = 28.3 -- значение соответствует 1 см
para_props.alignment = DocumentAPI.Alignment_Center
para_props.firstLineIndent = 28.3 -- значение соответствует 1 см
para_props.leftIndent = 28.3 -- значение соответствует 1см
para_props.lineSpacing = DocumentAPI.LineSpacing(5.0,
DocumentAPI.LineSpacingRule_Multiple)
para_props.rightIndent = 28.3 -- значение соответствует 1см
--
para:setParagraphProperties(para_props)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()


for para in paragraphs:enumerate() do
   local para_props = para:getParagraphProperties()
   para_props.afterSpacing = 28.3 -- значение соответствует 1 см
   para_props.beforeSpacing = 28.3 -- значение соответствует 1 см
   para_props.alignment = DocumentAPI.Alignment_Center
   para_props.firstLineIndent = 28.3 -- значение соответствует 1 см
   para_props.leftIndent = 28.3 -- значение соответствует 1см
   para_props.lineSpacing = DocumentAPI.LineSpacing(5.0, DocumentAPI.LineSpaci
ngRule_Multiple)
   para_props.rightIndent = 28.3 -- значение соответствует 1см
   para:setParagraphProperties(para_props)
end
```

3.5.8 Таблица DocumentAPI.LineSpacing

Таблица DocumentAPI.LineSpacing задает межстрочный интервал абзаца. Поля таблицы приведены в Таблице 33. Для управления значением межстрочного интервала используются значения, представленные в разделе `DocumentAPI.LineSpacingRule`.

Таблица 33 – Параметры межстрочного интервала

Поле	Описание
`DocumentAPI.LineSpacing.value`	Значение межстрочного интервала.
`DocumentAPI.LineSpacing.rule`	Правило формирования межстрочного интервала `DocumentAPI.LineSpacingRule`.

Пример:

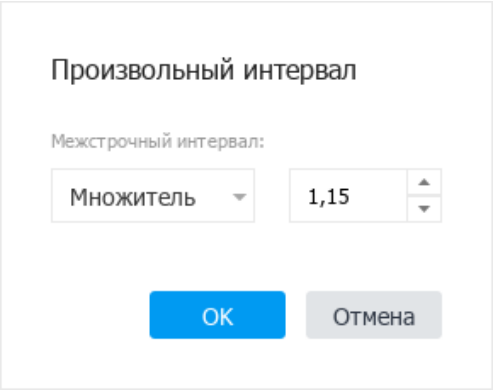
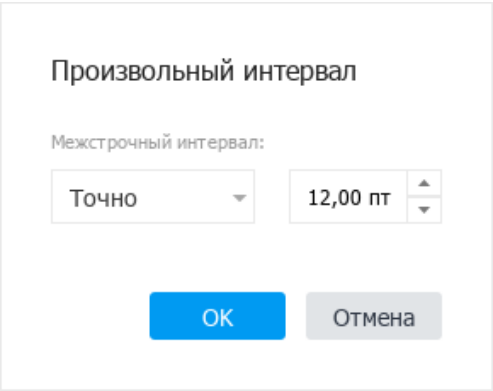
```
-- Конструктор
local lineSpacing = DocumentAPI.LineSpacing(1.5, DocumentAPI.LineSpacingRule_
Multiple)
-- Обращение к полям
lineSpacing.value = 1
lineSpacing.rule = DocumentAPI.LineSpacingRule_Exact
```

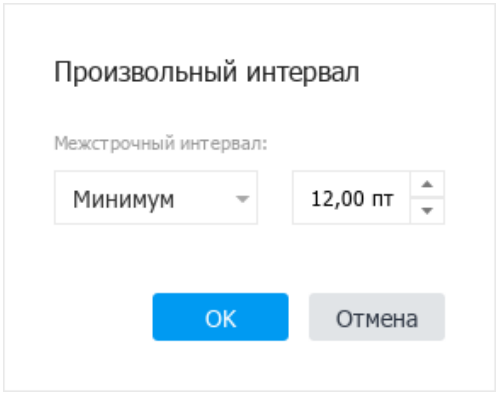
3.5.9 Таблица DocumentAPI.LineSpacingRule

В Таблице 34 представлены варианты правил формирования межстрочного интервала текстового абзаца.

Таблица 34 – Виды межстрочного интервала

Наименование константы	Описание
`DocumentAPI.LineSpacingRule_Multiple`	Установка произвольного межстрочного интервала с использованием множителя. При вызове необходимо указать значение множителя, например: `pPr.lineSpacing = DocumentAPI.LineSpacing(1.15, DocumentAPI.LineSpacingRule_Multiple)` В данном примере используется значение множителя **1.15**. Действие команды аналогично ручной настройке межстрочного интервала в диалоге «**Произвольный интервал**» (см. документ «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).

Наименование константы	Описание
	
`DocumentAPI.LineSpacingRule_Exact`	Установка произвольного межстрочного интервала с использованием точного значения. При вызове необходимо указать точное значение, например: `pPr.lineSpacing = DocumentAPI.LineSpacing(12.0, DocumentAPI.LineSpacingRule_Exact)` В данном примере используется точное значение **12.0**. Действие команды аналогично ручной настройке межстрочного интервала в диалоге «**Произвольный интервал**» 
`DocumentAPI.LineSpacingRule_AtLeast`	Установка произвольного межстрочного интервала с использованием минимального значения. При вызове необходимо указать минимальное значение, например: `pPr.lineSpacing = DocumentAPI.LineSpacing(12.0, DocumentAPI.LineSpacingRule_AtLeast)` В данном примере используется точное значение **12.0**. Действие команды аналогично ручной настройке межстрочного интервала в диалоге «**Произвольный интервал**» (см. документ «МойОфис Стандартный. МойОфис Текст. Руководство пользователя»).

Наименование константы	Описание
	

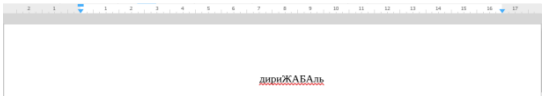
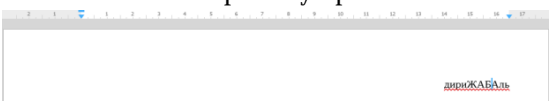
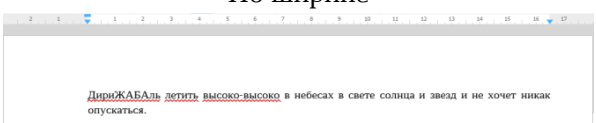
Пример:

```
p = document:getBlocks():getParagraph(0)
pPr = p:getParagraphProperties()
pPr.lineSpacing = DocumentAPI.LineSpacing(5.0,
DocumentAPI.LineSpacingRule_Multiple)
p:setParagraphProperties(pPr)
```

3.5.10 Таблица DocumentAPI.Alignment

В Таблице 35 представлены варианты выравнивания текста по горизонтали в текстовом редакторе или содержимого ячеек в табличном редакторе.

Таблица 35 – Варианты выравнивания по горизонтали

Наименование константы	Описание
DocumentAPI.Alignment_Default	Выравнивание по умолчанию.
DocumentAPI.Alignment_Left	По левому краю 
DocumentAPI.Alignment_Center	По центру 
DocumentAPI.Alignment_Right	По правому краю 
DocumentAPI.Alignment_Justify	По ширине 

Пример:

```
local para = document:getBlocks():getParagraph(0)
local props = para:getParagraphProperties()
props.alignment = DocumentAPI.Alignment_Center
para:setParagraphProperties(props)
```

3.6 Рецензирование документов

Средства рецензирования документа доступны в текстовом редакторе, они позволяют выполнять следующие действия:

– помечать изменения, вносимые пользователем в текстовый документ (DocumentAPI.TrackedChange);
– ассоциировать текстовый комментарий с фрагментом текстового документа (DocumentAPI.Comments).

Данные механизмы используются на стадии рецензирования или согласования документа с последующим внесением замечаний. Функции объектной модели для работы со средствами рецензирования позволяют получить детальную информацию о каждом изменении: автор изменения, дата внесения изменения, оригинальный текст, измененный текст.

Для включения или отключения режима рецензирования используется метод document:setChangesTrackingEnabled(). Для проверки текущего статуса данного режима используется метод document:isChangesTrackingEnabled().

Пример:

```
document:setChangesTrackingEnabled(true)
print(document:isChangesTrackingEnabled())
```

Инструменты рецензирования применяются к диапазону документа, по этой причине методы доступа к ним находятся в таблице DocumentAPI.Range (см. Рисунок 1123.1).

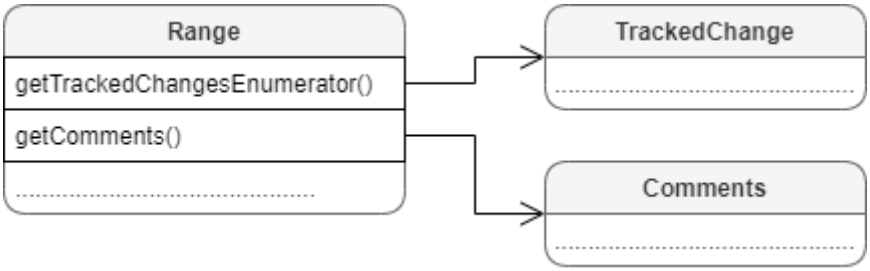


Рисунок 1123.1 – Инструменты рецензирования документа

3.6.1 Таблица DocumentAPI.TrackedChange

Таблица `DocumentAPI.TrackedChange` представляет отслеживаемое изменение в диапазоне документа (см. Рисунок 1123.2).

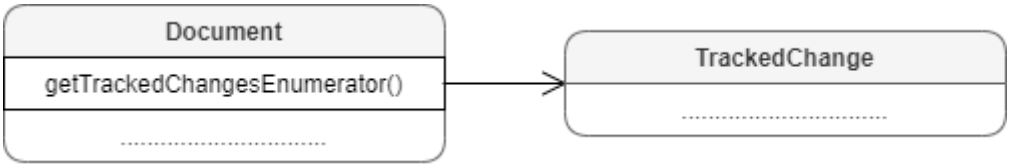


Рисунок 1123.2 – Объектная модель таблиц для работы с отслеживаемыми изменениями

Для получения списка отслеживаемых изменений используется метод `Range.enumerateTrackedChanges()`.

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()
for change in changesList do
  print(change:getRange():extractText())
end
```

3.6.1.1 Метод TrackedChange:getRange

Метод возвращает объект `DocumentAPI.Range`, который соответствует измененному диапазону внутри абзаца.

Пример:

```
local tracked_changes = document:getRange():enumerateTrackedChanges()
for tracked_change in tracked_changes do
  print(tracked_change:getRange():extractText())
end
```

3.6.1.2 Метод TrackedChange:getType

Метод позволяет получить информацию о типе отслеживаемого изменения DocumentAPI.TrackedChangeType.

Пример:

```
local tracked_changes = document:getRange():enumerateTrackedChanges()
for tracked_change in tracked_changes do
  print(tracked_change:getType())
end
```

3.6.1.3 Метод TrackedChange:getInfo

Метод позволяет получить информацию об отслеживаемых изменениях (DocumentAPI.TrackedChangeInfo).

Пример:

```
local tracked_changes = document:getRange():enumerateTrackedChanges()
for tracked_change in tracked_changes do
  print(tracked_change:getInfo().author.name)
end
```

3.6.2 Таблица DocumentAPI.TrackedChangeInfo

Таблица DocumentAPI.TrackedChangeInfo содержит информацию об отслеживаемых изменениях. Описание полей таблицы представлено в Таблице 36.

Таблица 36 – Описание полей таблицы DocumentAPI.TrackedChangeInfo

Поле	Тип	Описание
DocumentAPI.TrackedChangeInfo.author	UserInfo	Автор изменений.
DocumentAPI.TrackedChangeInfo.timeStamp	DateTime	Дата и время изменений .

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()
for change in changesList do
  local trackedChangeInfo = change:getInfo()
  local author = trackedChangeInfo.author
  local ts = trackedChangeInfo.timeStamp
  local ts_msg = string.format("%d/%d/%d - %d:%d:%d", ts.day, ts.month, ts.ye
ar, ts.hour, ts.minute, ts.second)
  print(author.name, ts_msg)
end
```

3.6.2.1 Метод TrackedChangeInfo:__eq

Метод используется для определения эквивалентности двух отслеживаемых изменений.

3.6.2.2 Метод TrackedChangeInfo:__ne

Метод используется для определения неэквивалентности двух отслеживаемых изменений.

3.6.3 Таблица DocumentAPI.DateTime

Таблица `DocumentAPI.DateTime` предоставляет дату и время с точностью до секунды. Используется для поля `TrackedChangeInfo.timeStamp`. Описание полей таблицы `DocumentAPI.DateTime` представлено в Таблице 37.

Таблица 37 – Описание полей таблицы `DocumentAPI.DateTime`

Поле	Тип	Описание
`DocumentAPI.DateTime.year`	Дата	Год
`DocumentAPI.DateTime.month`	Дата	Месяц
`DocumentAPI.DateTime.day`	Дата	День
`DocumentAPI.DateTime.hour`	Время	Часы
`DocumentAPI.DateTime.minute`	Время	Минуты
`DocumentAPI.DateTime.second`	Время	Секунды

3.6.3.1 Метод DateTime:__eq

Метод используется для определения эквивалентности двух значений времени.

3.6.3.2 Метод DateTime:__ne

Метод используется для определения неэквивалентности двух значений времени.

3.6.4 Таблица DocumentAPI.TrackedChangeType

Типы отслеживаемых изменений представлены в Таблице 38.

Таблица 38 – Типы отслеживаемых изменений

Наименование константы	Описание
`DocumentAPI.TrackedChangeType_Added`	Добавленные изменения.
`DocumentAPI.TrackedChangeType_Removed`	Удаленные изменения.

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()
for change in changesList do
  if DocumentAPI.TrackedChangeType_Added == change:getType() then action = "Добавлено: " else action = "Удалено: " end
  print(action)
end
```

3.6.5 Таблица DocumentAPI.Comments

Таблица `DocumentAPI.Comments` содержит коллекцию комментариев диапазона (см. Рисунок 1124).

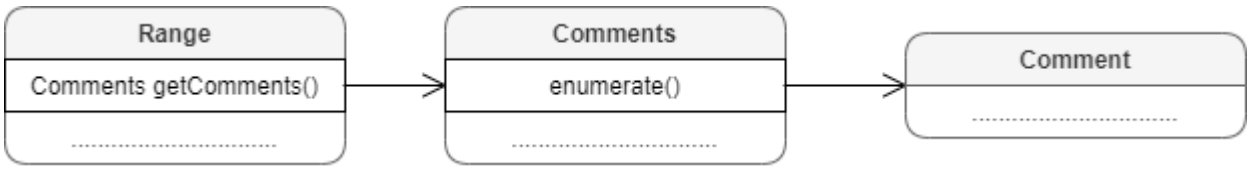


Рисунок 1124 – Объектная модель таблиц для работы с комментариями

Для получения списка комментариев используется метод `Range.getComments()`.

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    local commentInfo = comment:getInfo()
    local name = commentInfo.author.name
    print("Комментарий " .. name .. ": ", comment:getText())
end
```

3.6.5.1 Метод Comments:enumerate

Метод возвращает коллекцию комментариев всего документа.

Пример:

```
local comments = document:getComments()
for comment in comments:enumerate() do
  print(comment:getText())
end
```

3.6.6 Таблица DocumentAPI.Comment

Таблица `DocumentAPI.Comment` предоставляет доступ к следующим свойствам комментария:

- диапазон текста `DocumentAPI.Range`, который описывает комментарий;
- текст комментария;
- информация о комментарии `DocumentAPI.TrackedChangeInfo`;
- признак того, что комментарий принят;
- список ответов на комментарий `DocumentAPI.Comments`.

3.6.6.1 Метод Comment:getRange

Метод возвращает диапазон документа `DocumentAPI.Range`, которому соотвествует комментарий.

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    print("Диапазон комментария: ", comment:getRange():extractText())
end
```

3.6.6.2 Метод Comment:getText

Метод возвращает текст комментария.

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    print("Текст комментария: ", comment:getText())
end
```

3.6.6.3 Метод Comment:getInfo

Метод предоставляет доступ к информации о комментарии `DocumentAPI.TrackedChangeInfo` (автор изменения, дата и т. д).

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    local commentInfo = comment:getInfo()
    local name = commentInfo.author.name
```

```
    print("Автор комментария:", name)
end
```

3.6.6.4 Метод Comment:isResolved

Метод возвращает значение `true`, если комментарий принят.

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    print("Комментарий принят:", comment:isResolved())
end
```

3.6.6.5 Метод Comment:getReplies

Метод предоставляет доступ к ответам на комментарии. Ответы находятся в такой же таблице `DocumentAPI.Comments`, как и сами комментарии документа.

Пример:

```
local commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    local commentReplies = comment:getReplies()
    for reply in commentReplies:enumerate() do
        local name = reply.author.name
        print("Ответ на комментарий " .. name .. ": ", reply:getText())
    end
end
```

3.7 Работа с закладками

Основной таблицей для работы с закладками является `DocumentAPI.Bookmarks`. Список закладок документа возвращает метод `document:getBookmarks()`. Метод `Bookmarks.getBookmarkRange()` возвращает диапазон текста, метод `Bookmarks.removeBookmark()` удаляет закладку по имени. Для создания закладки используется метод `Position.insertBookmark()`.

Доступны следующие операции с закладками:

- вставка закладки в указанное местоположение;
- удаление закладки с заданным именем;
- поиск закладки по имени;
- замена текстового содержимого закладки;
- вставка текста в закладку;

- удаление содержимого закладки;
- получение текстового содержимого закладки;
- вставка таблицы в закладку.

Вставка закладки в указанное местоположение

```
-- Вставка новой закладки с именем Signers в начало документа
local sig_pos = document:getRange():getBegin()
sig_pos:insertBookmark("Signers")
```

Удаление закладки с заданным именем

```
-- Удаление закладки "Signers"
document:getBookmarks():removeBookmark("Signers")
```

Поиск закладки по имени

```
-- Поиск закладки "Signers" по имени
local sig_rng = document:getBookmarks():getBookmarkRange("Signers")
```

Замена текстового содержимого закладки

```
-- Замена содержимого закладки на текст "Lua"
local bookmarks = document:getBookmarks()
local bookmarkRange = bookmarks:getBookmarkRange( "bm_1" )
bookmarkRange.replaceText("Lua")
```

Вставка текста в закладку

```
sig_rng:getBegin():insertText("Лист")
```

Удаление содержимого закладки

```
sig_rng:removeContent()
```

Получение текстового содержимого закладки

```
local msg = sig_rng:extractText()
print(msg)
```

Вставка таблицы в закладку

```
-- Вставка таблицы в закладку "Signers"
local tbl_id = sig_rng:getEnd():insertTable(3, 3, "signers_list")
```

3.7.1 Таблица DocumentAPI.Bookmarks

Предоставляет доступ к операциям с закладками в документе.

3.7.1.1 Метод Bookmarks:getBookmarkRange

Возвращает объект DocumentAPI.Range для дальнейшей работы с содержимым закладки (bookmark).

Пример:

```
local bookmarks = document:getBookmarks()
local bookmark = bookmarks:getBookmarkRange("Bookmark")
bookmark:replaceText("Lua")
```

3.7.1.2 Метод Bookmarks:removeBookmark

Удаляет закладку по ее названию.

Пример:

```
document:getBookmarks():removeBookmark("Bookmark")
```

3.8 Таблицы и ячейки

3.8.1 Доступ к таблицам

Доступ к таблицам DocumentAPI.Table осуществляется из DocumentAPI.Blocks (см. Рисунок 1125). В табличном документе таблицами являются листы документа.

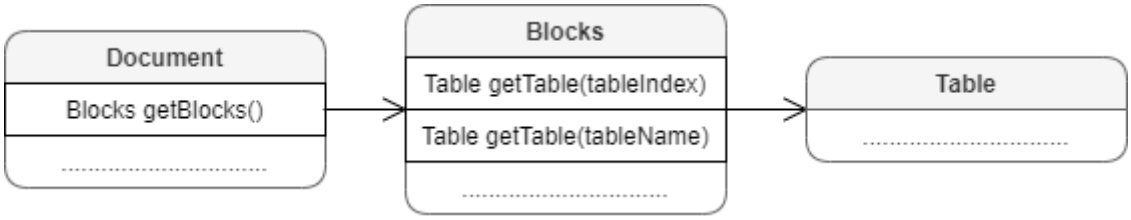


Рисунок 1125 – Объектная модель для работы с таблицами

Для получения таблицы используется метод Blocks:getTable(). В качестве аргумента используется индекс или имя таблицы.

Примеры:

```
local table = document:getBlocks():getTable(0)

local table = document:getBlocks():getTable("Таблица1")
```

Для перечисления таблиц текстового документа или листов табличного документа можно использовать метод Blocks:enumerateTables().

```
for sheet in document:getBlocks():enumerateTables() do
    print(sheet:getName())
end
```

Для табличного документа также доступен вариант перечисления листов документа посредством использования метода `Blocks:enumerate()` с дальнейшим преобразованием блока в таблицу.

```
for sheet in document:getBlocks():enumerate() do
    print(sheet:toTable():getName())
end
```

Для вставки таблицы в текстовый документ или листа в табличный документ используется метод `Position:insertTable()`. В качестве аргументов передаются размеры и имя таблицы.

Пример:

```
local position = document:getRange():getEnd()
position:insertTable(4, 3, "Таблица1")
```

В табличном документе таблицы являются листами документа. Для листов табличного документа доступны следующие операции:

– добавление листа, см. описание метода `Position.insertTable()`;
– переименование листа, см. описание метода `Table.setName()`;
– копирование и перемещение листа, см. описание метода `Table.duplicate()`;
– удаление листа, см. описание метода `Table.remove()`;
– перечисление листов документа, варианты реализации, см. варианты реализации в разделе Таблицы и ячейки;
– обращение к листу документа, см. варианты реализации в разделе Таблицы и ячейки;
– скрытие и отображение листов, см. описание метода `Table.setVisible()`;

3.8.2 Доступ к ячейкам

Доступ к ячейкам таблицы возможен двумя способами (см. Рисунок 1126):

– непосредственно из таблицы, используя метод `Table.getCell()`;
– из диапазона ячеек методом перечисления `CellRange.enumerate()`.

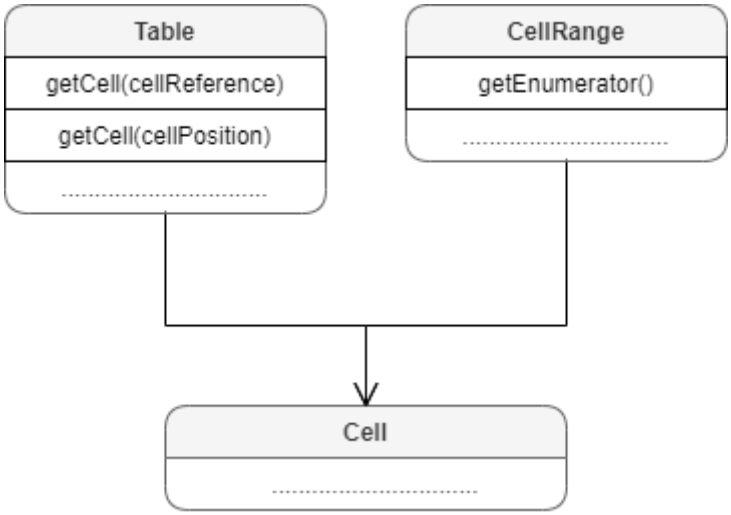


Рисунок 1126 – Объектная модель для работы с ячейками таблиц

Для получения содержимого ячейки, заполнения данных, а также для форматирования ячейки используется объект `DocumentAPI.Cell`, представляющий ячейку таблицы с указанным адресом. Метод `Table.getCell()` возвращает экземпляр таблицы `Cell`.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")
```

Второй вариант доступа к ячейке - перечисление диапазона ячеек методом `CellRange.enumerate()`.

Пример:

```
local table = document:getBlocks():getTable(0)
local rng = table:getCellRange("B3:C4")
for cell in rng:enumerate() do
    print(cell:getFormattedValue())
end
```

Для установки значений ячеек используются методы `Cell:setText`, `Cell:setNumber`, `Cell:setFormula`, `Cell:setBool`.

Примеры:

```
local sheet = document:getBlocks():getTable("Лист2")
```

```
--setText, текстовое значение
sheet:getCell("A1"):setText("Текст")

--setNumber, числовое значение с фиксированной точкой
sheet:getCell("B2"):setNumber(10)

--setNumber, числовое значение с плавающей точкой
sheet:getCell("B3"):setNumber(1,0)

--setFormula, текст формулы
sheet:getCell("B4"):setFormula("=SUM(B2:B3)")

--setBool, логическое значение
sheet:getCell("B4"):setBool(false)
```

Для установки даты и времени используется функция Cell:setFormattedValue. Данная функция пытается определить тип значения, переданного в качестве аргумента (число, дата и т.д.) и применяет необходимое форматирование.

Пример:

```
local sheet = document:getBlocks():getTable("Лист1")

--setFormattedValue, дата
sheet:getCell("B5"):setFormattedValue("22.07.2020")

--setFormattedValue, время
sheet:getCell("B6"):setFormattedValue("12:39")
```

При необходимости есть возможность явно указать формат вводимого значения DocumentAPI.CellFormat (процентный, денежный, экспоненциальный и т.д.), для этого используется функция Cell.SetFormat().

Пример:

```
local sheet = document:getBlocks():getTable("Лист1")
local value = 12
local cell = sheet:getCell("B1")
-- Установка формата данных
cell:setFormat(DocumentAPI.CellFormat_Accounting)
cell:setNumber(value)
```

Для получения значения ячейки используется метод `Cell.getFormattedValue()`.

Пример:

```
local sheet = document:getBlocks():getTable("Лист1")
local value = sheet:getCell("B1"):getFormattedValue()
```

3.8.2.1 Таблица DocumentAPI.CellFormat

По умолчанию при создании документа всем ячейкам присваивается формат «Общий». Полный список форматов представлен в Таблице 39.

Таблица 39 – Поддерживаемые форматы ячеек таблицы

Наименование константы	Описание
`DocumentAPI.CellFormat_General`	Формат ячейки «Общий». В этом формате в ячейке отображаются первые 9 символов числа, остальные доступны для просмотра в строке формул. Для дробных чисел в формате «Общий» незначащие нули в дробной части не отображаются. Числа, состоящие более чем из 12 символов, переводятся в экспоненциальную форму после завершения ввода в ячейку.
`DocumentAPI.CellFormat_Percentage`	Формат ячейки «Процентный». Этот формат используется для представления чисел как процентов. При применении формата «Процентный» введенное число умножается на 100 и обозначается знаком «%».
`DocumentAPI.CellFormat_Number`	Формат ячейки «Числовой». Если в ячейке с форматом «Числовой» содержится дробное число, то можно указать количество знаков, отображаемых в данном числе после разделителя.
`DocumentAPI.CellFormat_Text`	Формат ячейки «Текстовый».
`DocumentAPI.CellFormat_Currency`	Формат ячейки «Денежный». Этот формат используется для представления чисел со знаком или кодом валюты.

Наименование константы	Описание
DocumentAPI.CellFormat_Accounting	Формат ячейки «Финансовый». Этот формат применяется для чисел, используемых в бухгалтерских документах. В формате «Финансовый» введенное число автоматически дополняется названием валюты, которая соответствует настройкам системы компьютера. Отрицательные числа в формате «Финансовый» заключаются в круглые скобки в ячейке, а в строке формул остаются в том виде, в котором они были введены.
DocumentAPI.CellFormat_Date	Формат ячейки «Дата». Этот формат автоматически присваивается числам, введенным в определенном виде, например, ДД.ММ.ГГГГ.
DocumentAPI.CellFormat_Time	Формат ячейки «Время». Этот формат автоматически присваивается числам, введенным в определенном виде, например, ЧЧ:ММ.
DocumentAPI.CellFormat_Fraction	Формат ячейки «Дробный». Этот формат используется для представления дробных чисел в виде обыкновенных дробей (то есть дробная часть заменяется на числитель и знаменатель).
DocumentAPI.CellFormat_Scientific	Формат ячейки «Экспоненциальный». Экспоненциальный (или научный) формат используется для представления больших чисел в короткой форме. Все введенные числа длиной более 12 символов автоматически переводятся в этот формат. В ячейке число в формате «Экспоненциальный» представлено следующим образом: – целая часть, всегда состоящая из одной цифры; – разделитель целой и дробной части; – дробная часть, по умолчанию состоящая из двух цифр; – показатель степени числа 10 в виде **E<знак**

Наименование константы	Описание
	показателя степени> <показатель степени>.
`DocumentAPI.CellFormat_Custom`	Пользовательский формат.

Использование данных констант позволяет установить выбранный формат. При этом будет использованы параметры формата по умолчанию.

Примеры использования:

```
local table = document:getBlocks():getTable(0)
local cell_B1 = table:getCell("B1")
cell_B1:setFormat(DocumentAPI.CellFormat_General)
local cell_B2 = table:getCell("B2")
cell_B2:setFormat(DocumentAPI.CellFormat_Percentage)
local cell_B3 = table:getCell("B3")
cell_B3:setFormat(DocumentAPI.CellFormat_Number)
```

Результат выполнения данного примера приведен на Рисунке 1127.

	A	B
1	CellFormat.General	1
2	CellFormat.Percentage	100,00%
3	CellFormat.Number	1,00

Рисунок 1127 – Результат установки формата

Пример форматирования ячейки также приведен в разделе, описывающем установку значений ячеек.

3.8.2.2 Таблица DocumentAPI.AccountingCellFormatting

Таблица содержит параметры финансового формата ячеек таблицы и используется в качестве аргумента метода `Cell:setFormat()`.

Описание полей таблицы `DocumentAPI.AccountingCellFormatting` представлено в Таблице 40.

Таблица 40 – Описание полей таблицы `DocumentAPI.AccountingCellFormatting`

Поле	Описание
`DocumentAPI.AccountingCellFormatting.decimalPlaces`	Количество десятичных позиций.
`DocumentAPI.AccountingCellFormattin`	Символ денежной единицы.

Поле	Описание
g.symbol	
DocumentAPI.AccountingCellFormatting.localeCode	Идентификатор кода языка (MS-LCID).
DocumentAPI.AccountingCellFormatting.fillSymbol	Символ заполнения.
DocumentAPI.AccountingCellFormatting.useThousandsSeparator	Использовать разделитель для тысячных.
DocumentAPI.AccountingCellFormatting.currencySignPlacement	Тип размещения знака валюты CurrencySignPlacement.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

cell:setFormat(DocumentAPI.CellFormat_Accounting)
local accountingCellFormatting = DocumentAPI.AccountingCellFormatting()
accountingCellFormatting.decimalPlaces = 3
accountingCellFormatting.symbol = 'Руб'

cell:setFormat(accountingCellFormatting)
print(cell:getFormattedValue())
```

3.8.2.3 Таблица DocumentAPI.PercentageCellFormatting

Содержит параметр для процентного формата ячеек таблицы, используется в качестве аргумента метода Cell:setFormat(). Описание полей таблицы DocumentAPI.PercentageCellFormatting представлено в Таблице 41.

Таблица 41 – Описание полей таблицы DocumentAPI.PercentageCellFormatting

Поле	Описание
DocumentAPI.PercentageCellFormatting.decimalPlaces	Количество десятичных позиций.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

local percentageCellFormatting = DocumentAPI.PercentageCellFormatting()
percentageCellFormatting.decimalPlaces = 2
```

```
cell:setFormat(percentageCellFormatting)
print(cell:getFormattedValue())
```

3.8.2.4 Таблица DocumentAPI.NumberCellFormatting

Таблица содержит параметры для числового формата ячеек таблицы, используется в качестве аргумента метода `Cell:setFormat()`. Описание полей таблицы `DocumentAPI.NumberCellFormatting` представлено в Таблице 42.

Таблица 42 – Описание полей таблицы `DocumentAPI.NumberCellFormatting`

Поле	Описание
`DocumentAPI.NumberCellFormatting.decimalPlaces`	Количество десятичных позиций.
`DocumentAPI.NumberCellFormatting.useThousandsSeparator`	Использовать разделитель для тысячных.
`DocumentAPI.NumberCellFormatting.useRedForNegative`	Использовать красный цвет для отрицательных значений.
`DocumentAPI.NumberCellFormatting.useBracketsForNagative`	Использовать скобки для отрицательных значений.
`DocumentAPI.NumberCellFormatting.hideSign`	Скрывать знак «минус» для отрицательных значений.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A2")

local numberCellFormatting = DocumentAPI.NumberCellFormatting()
numberCellFormatting.decimalPlaces = 2
numberCellFormatting.useThousandsSeparator = true
numberCellFormatting.useRedForNegative = true
numberCellFormatting.useBracketsForNegative = true
numberCellFormatting.hideSign = false

cell:setFormat(numberCellFormatting)
print(cell:getFormattedValue())
```

3.8.2.5 Таблица DocumentAPI.CurrencyCellFormatting

Таблица содержит параметры для денежного формата ячеек таблицы. Описание полей таблицы `DocumentAPI.CurrencyCellFormatting` представлено в Таблице 43.

Таблица 43 – Описание полей таблицы `DocumentAPI.CurrencyCellFormatting`

Поле	Описание
`DocumentAPI.CurrencyCellFormatting.decimalPlaces`	Количество десятичных позиций.
`DocumentAPI.CurrencyCellFormatting.symbol`	Символ денежной единицы.
`DocumentAPI.CurrencyCellFormatting.localeCode`	Идентификатор кода языка (MS-LCID).
`DocumentAPI.CurrencyCellFormatting.useThousandsSeparator`	Использовать разделитель для тысячных.
`DocumentAPI.CurrencyCellFormatting.useRedForNegative`	Использовать красный цвет для отрицательных значений.
`DocumentAPI.CurrencyCellFormatting.useBracketsForNegative`	Использовать скобки для отрицательных значений.
`DocumentAPI.CurrencyCellFormatting.hideSign`	Скрывать знак «минус» для отрицательных значений.
`DocumentAPI.CurrencyCellFormatting.currencySignPlacement`	Варианты размещения знака валюты `CurrencySignPlacement`.

Данная таблица используется в качестве аргумента метода `Cell:setFormat()`, см. пример.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

local currencyCellFormatting = DocumentAPI.CurrencyCellFormatting()
currencyCellFormatting.decimalPlaces = 2
currencyCellFormatting.useThousandsSeparator = true
currencyCellFormatting.useRedForNegative = true
currencyCellFormatting.useBracketsForNegative = true
currencyCellFormatting.hideSign = false
currencyCellFormatting.currencySignPlacement = DocumentAPI.CurrencySignPlacement_Suffix

cell:setFormat(currencyCellFormatting)
print(cell:getFormattedValue())
```

3.8.2.6 Таблица DocumentAPI.CurrencySignPlacement

Варианты размещения знака валюты представлены в Таблице 44. Данный тип используется в поле `currencyFormat` таблицы `DocumentAPI.LocaleInfo`, а также в поле `currencySignPlacement` таблицы `DocumentAPI.CurrencyCellFormatting` (см.пример в ее описании).

Таблица 44 – Описание полей таблицы `DocumentAPI.CurrencySignPlacement`

Поле	Описание	Пример
`DocumentAPI.CurrencySignPlacement_Prefix`	Размещение знака валюты до значения.	$12.00
`DocumentAPI.AccountingCellFormatting_Suffix`	Размещение знака валюты после значения.	12,00 ₽

3.8.2.7 Таблица DocumentAPI.DateTimeCellFormatting

Таблица содержит параметры для формата ячеек таблицы типа Дата и Время, используется в качестве аргумента метода `Cell:setFormat()`. Описание полей таблицы `DocumentAPI.DateTimeCellFormatting` представлено в Таблице 45.

Таблица 45 – Описание полей таблицы `DocumentAPI.DateTimeCellFormatting`

Поле	Описание
`DocumentAPI.DateTimeCellFormatting.dateListID`	Формат даты `DatePatterns`.
`DocumentAPI.DateTimeCellFormatting.timeListID`	Формат времени `TimePatterns`.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")

local dateTimeCellFormatting = DocumentAPI.DateTimeCellFormatting()
dateTimeCellFormatting.dateListID = DocumentAPI.DatePatterns_FullDate
dateTimeCellFormatting.timeListID = DocumentAPI.TimePatterns_LongTime

cell:setFormat(dateTimeCellFormatting)
print(cell:getFormattedValue())
```

3.8.2.8 Таблица DocumentAPI.DatePatterns

Форматы даты представлены в Таблице 46. Пример использования см. в главе `DocumentAPI.DateTimeCellFormatting`.

Таблица 46 – Форматы даты

Наименование константы	Описание
`DocumentAPI.DatePatterns_DayMonthTextLongYearLong`	'mmmm dd, yyyy' для языка en_US.
`DocumentAPI.DatePatterns_FullDate`	'день недели, mmmm dd, yyyy' для языка en_US.
`DocumentAPI.DatePatterns_DayMonthNumberLongYearLong`	'mm/dd/yyyy' для языка en_US.
`DocumentAPI.DatePatterns_DayMonthNumberLongYearShort`	'm/dd/yy' для языка en_US.
`DocumentAPI.DatePatterns_DayMonthNumberShortYearShort`	'dd-mmm' для языка en_US.
`DocumentAPI.DatePatterns_DayMonthTextShort`	'mmm-yy' для языка en_US.
`DocumentAPI.DatePatterns_DayMonthTextShortYearShort`	'mmm dd, yy' для языка en_US.
`DocumentAPI.DatePatterns_DayMonthYearLongNonLocalizableHyphen`	Нелокализуемый шаблон 'dd-mm-yyyy'.
`DocumentAPI.DatePatterns_DayMonthYearLongNonLocalizableSlash`	Нелокализуемый шаблон 'dd/mm/yyyy'.

3.8.2.9 Таблица DocumentAPI.TimePatterns

Форматы времени представлены в Таблице 47. Пример использования см. в главе `DocumentAPI.DateTimeCellFormatting`.

Таблица 47 – Форматы времени

Наименование константы	Описание
`DocumentAPI.TimePatterns_ShortTime`	'hh:mm AM/PM' для языка en_US.
`DocumentAPI.TimePatterns_LongTime`	'hh:mm:ss AM/PM' для языка en_US.

3.8.2.10 Таблица DocumentAPI.FractionCellFormatting

Таблица содержит параметры для дробного формата ячеек таблицы. Данная таблица используется в качестве аргумента метода `Cell:setFormat()`. Описание полей таблицы `DocumentAPI.FractionCellFormatting` представлено в Таблице 48.

Таблица 48 – Описание полей таблицы `DocumentAPI.FractionCellFormatting`

Поле	Описание
`DocumentAPI.FractionCellFormatting.minNumeratorDigits`	Количество позиций числителя.
`DocumentAPI.FractionCellFormatting.minDenominatorDigits`	Количество позиций знаменателя.
`DocumentAPI.FractionCellFormatting denominatorValue`	Знаменатель.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("B2")

local fractionCellFormatting = DocumentAPI.FractionCellFormatting()
fractionCellFormatting.minNumeratorDigits = 2
fractionCellFormatting.minDenominatorDigits = 3
fractionCellFormatting.denominatorValue = 22

cell:setFormat(fractionCellFormatting)
print(cell:getFormattedValue())
```

3.8.2.11 Таблица DocumentAPI.ScientificCellFormatting

Таблица содержит параметры для экспоненциального формата ячеек таблицы. Данная таблица используется в качестве аргумента метода `Cell:setFormat()`. Описание полей таблицы `DocumentAPI.ScientificCellFormatting` представлено в Таблице 49.

Таблица 49 – Описание полей таблицы `DocumentAPI.ScientificCellFormatting`

Поле	Описание
`DocumentAPI.ScientificCellFormatting.decimalPlaces`	Количество десятичных позиций.
`DocumentAPI.ScientificCellFormatting.minExponentDigits`	Минимальное количество позиций экспоненты.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("B2")

local scientificCellFormatting = DocumentAPI.ScientificCellFormatting()
scientificCellFormatting.decimalPlaces = 2
```

```
scientificCellFormatting.minExponentDigits = 3


cell:setFormat(scientificCellFormatting)
print(cell:getFormattedValue())
```

3.8.3 Форматирование ячеек

При работе с ячейками таблиц можно использовать следующие варианты форматирования:

– форматирование параметров ячейки DocumentAPI.CellProperties, например, цвет фона, угол поворота текста;
– форматирование абзаца ячейки, например, отступы абзаца, межстрочный интервал текста;
– форматирование текста, например, цвет текста, начертание;
– задание параметров границ ячеек.

Содержимое ячейки (контент), вне зависимости от того является ли оно текстом, числовым значением или формулой, также описывается экземпляром класса Paragraph, и обладает свойствами DocumentAPI.ParagraphProperties. Это дает возможность управлять настройками отображения контента как отдельного абзаца, так и группы абзацев (например, если ячейка содержит несколько предложений текста). Для управления этим настройками используются методы Cell.getParagraphProperties() и Cell.setParagraphProperties().

Пример установки и получения свойств параграфа ячейки:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A2") --(DocumentAPI.CellPosition(1,0))


local paraProps = cell:getParagraphProperties()
paraProps.alignment = DocumentAPI.Alignment_Center
cell:setParagraphProperties(paraProps)
```

Управление настройками текста ячейки (шрифт, цвет) производится через соответствующий ему диапазон. Класс Cell позволяет получить диапазон для всего контента с помощью метода Cell.getRange(). Далее, метод Range.getTextProperties() позволяет получить экземпляр класса DocumentAPI.TextProperties, представляющий свойства текста. После изменения

значения свойств их необходимо применить к тексту ячейки с помощью метода Range.setTextProperties().

Пример настроек текста ячейки:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell(DocumentAPI.CellPosition(0,1))


local textProps = cell:getRange():getTextProperties()
textProps.bold = true
textProps.italic = true
local rgba = DocumentAPI.ColorRGBA(121,112,212,255)
textProps.textColor = DocumentAPI.Color(rgba)
cell:getRange():setTextProperties(textProps)
```

3.8.4 Форматирование границ ячеек

Для оформления границ ячеек используется таблица DocumentAPI.Borders (см. Рисунок 1128). Она описывает свойства полей, соответствующих границам и диагоналям ячейки: Left, Right, Top, Bottom, DiagonalDown, DiagonalUp, InnerHorizontal, InnerVertical. Каждая граница ячейки описывается таблицей DocumentAPI.LineProperties, которая, в свою очередь, обладает свойствами DocumentAPI.LineStyle, DocumentAPI.LineEndingProperties, DocumentAPI.Color, LineWidth.

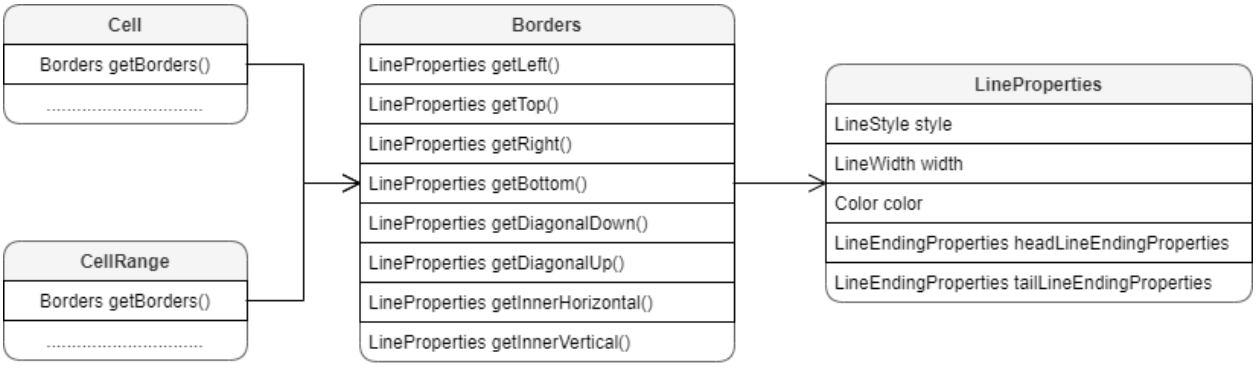


Рисунок 1128 – Таблицы для работы с границами ячеек

Для оформления границ отдельной ячейки или группы ячеек необходимо выполнить следующие действия:

– получить ячейку DocumentAPI.Cell или область ячеек DocumentAPI.CellRange;
– настроить параметры для рисования линии границы с помощью экземпляра класса DocumentAPI.LineProperties;
– настроить свойства линии: левой границы, верхней границы и т.д. с помощью экземпляра класса DocumentAPI.Borders;
– установить границы ячеек с помощью Cell.setBorders() или CellRange.setBorders().

Пример настройки границ ячеек:

```
local sheet = document:getBlocks():getTable("Лист2")
local cellRange = sheet:getCellRange("F3:H7")

--Настроить параметры для рисования линии
local lineProp = DocumentAPI.LineProperties()
lineProp.style = DocumentAPI.LineStyle_Solid
lineProp.width = 1.5
local lc = DocumentAPI.ColorRGBA(55, 146, 179, 200)
lineProp.color = DocumentAPI.Color(lc)

--Настроить положение линии - обводка по внешней границе области
local borders = DocumentAPI.RangeBorders()

--установка внешних границ
borders:setOuter(lineProp)

--Нарисовать границы области
cellRange:setBorders(borders)
```

3.8.4.1 Таблица DocumentAPI.Borders

Таблица DocumentAPI.Borders предназначена для оформления границ отдельной ячейки таблицы (см. Таблицу 50). Параметры линии, такие как тип линии, ее ширина и цвет, задаются с помощью таблицы DocumentAPI.LineProperties.

Таблица 50 – Описание методов таблицы DocumentAPI.Borders

Метод	Описание
Borders:setLeft	Установка левой границы ячейки.

Метод	Описание
Borders:setRight	Установка правой границы ячейки.
Borders:setTop	Установка верхней границы ячейки.
Borders:setBottom	Установка нижней границы ячейки.
Borders:setDiagonalDown	Установка диагональной линии. 
Borders:setDiagonalUp	Установка диагональной линии. 
Borders:setOuter	Установка внешних границ ячейки.
Borders:setDiagonals	Установка обеих типов диагональных линий одновременно.
Borders:setInnerHorizontal	Установка внутренних горизонтальных границ ячейки.
Borders:setInnerVertical	Установка внутренних вертикальных границ ячейки.
Borders:setInner	Установка внутренних границ ячейки.
Borders:setAll	Установка всех границ ячейки.
Borders:getLeft	Получение левой границы ячейки.
Borders:getRight	Получение правой границы ячейки.
Borders:getTop	Получение верхней границы ячейки.
Borders:getBottom	Получение нижней границы ячейки.
Borders:getDiagonalDown	Получение диагональной линии.
Borders:getDiagonalUp	Получение диагональной линии.
Borders:getInnerHorizontal	Получение внутренних горизонтальных границ ячейки.
Borders:getInnerVertical	Получение внутренних вертикальных границ ячейки.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("C3")


LineProperties = DocumentAPI.LineProperties()
LineProperties.style = DocumentAPI.LineStyle_Dash
LineProperties.width = 1.5
LineProperties.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0,
255))


borders = DocumentAPI.Borders()
borders = borders:setLeft(LineProperties)
```

```
borders = borders:setRight(LineProperties)
borders = borders:setTop(LineProperties)
borders = borders:setBottom(LineProperties)


borders = cell:setBorders(borders)
```

3.8.4.2 Таблица DocumentAPI.RangeBorders

Таблица `DocumentAPI.RangeBorders` оставлена для совместимости. Вместо нее необходимо использовать таблицу `DocumentAPI.Borders`.

3.8.4.3 Таблица DocumentAPI.LineProperties

Таблица `DocumentAPI.LineProperties` предназначена для установки таких параметров линии, как тип, ширина, цвет (см. Рисунок 1129).

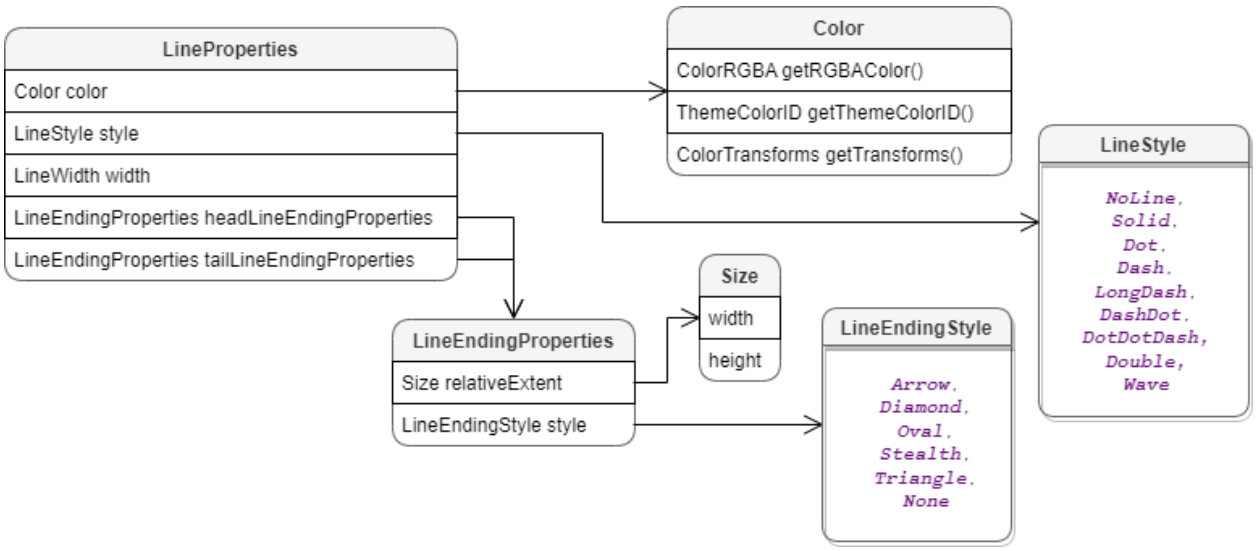


Рисунок 1129 – Свойства границ ячеек

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("C3")


lineProperties = DocumentAPI.LineProperties()
lineProperties.style = DocumentAPI.LineStyle_Solid
lineProperties.width = 1.5
lineProperties.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(55, 146, 179,
200))
```

```
borders = DocumentAPI.Borders()
borders = borders:setTop(lineProperties)
local brds = cell:setBorders(borders)
```

3.8.4.3.1 Поле LineProperties.style

Поле предназначено для установки типа линии. Допустимые значения представлены в разделе DocumentAPI.LineStyle.

3.8.4.3.2 Поле LineProperties.width

Поле предназначено для установки ширины линии. Тип - числовой.

3.8.4.3.3 Поле LineProperties.color

Поле предназначено для установки цвета линии. Тип - DocumentAPI.Color.

3.8.4.3.4 Поле LineProperties.headLineEndingProperties

Поле предназначено для оформления начала линии DocumentAPI.LineEndingProperties.

3.8.4.3.5 Поле LineProperties.tailLineEndingProperties

Поле предназначено для оформления конца линии DocumentAPI.LineEndingProperties.

3.8.4.4 Таблица DocumentAPI.LineEndingProperties

Таблица DocumentAPI.LineEndingProperties содержит варианты оформления окончаний линий. Описание полей таблицы DocumentAPI.LineEndingProperties представлено в Таблице 51. Используется в полях headLineEndingProperties и tailLineEndingProperties таблицы DocumentAPI.LineProperties.

Таблица 51 – Описание полей таблицы DocumentAPI.LineEndingProperties

Поле	Тип	Описание
DocumentAPI.LineEndingProperties.style	LineEndingStyle	Стиль окончания линии.
DocumentAPI.LineEndingProperties.relativeExtent	Size	Размер окончания линии относительно ее ширины .

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = tbl:getCell("C3")
```

```
lineProperties = DocumentAPI.LineProperties()
lineProperties.headLineEndingProperties = DocumentAPI.LineEndingProperties()
lineProperties.headLineEndingProperties.style =
DocumentAPI.LineEndingStyle_Arrow

lineProperties.headLineEndingProperties.relativeExtent = DocumentAPI.SizeU()
lineProperties.headLineEndingProperties.relativeExtent.width = 2
lineProperties.headLineEndingProperties.relativeExtent.height = 2

lineProperties.tailLineEndingProperties = DocumentAPI.LineEndingProperties()
lineProperties.tailLineEndingProperties.style =
DocumentAPI.LineEndingStyle_Arrow
lineProperties.tailLineEndingProperties.relativeExtent = DocumentAPI.SizeU()
lineProperties.tailLineEndingProperties.relativeExtent.width = 2
lineProperties.tailLineEndingProperties.relativeExtent.height = 2

borders = DocumentAPI.Borders()
borders = borders:setTop(lineProperties)
cell:setBorders(borders)
```

3.8.4.5 Таблица DocumentAPI.LineStyle

В Таблице 52 приведены типы линий. Используется в поле `style` таблицы `DocumentAPI.LineProperties`.

Таблица 52 – Типы линий

Наименование константы	Описание
`DocumentAPI.LineStyle_NoLine`	Нет линии
`DocumentAPI.LineStyle_Solid`	
`DocumentAPI.LineStyle_Dot`	
`DocumentAPI.LineStyle_Dash`	
`DocumentAPI.LineStyle_LongDash`	
`DocumentAPI.LineStyle_DashDot`	

Наименование константы	Описание
DocumentAPI.LineStyle_DotDotDash	
DocumentAPI.LineStyle_Double	
DocumentAPI.LineStyle_Wave	

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("C3")

lineProperties = DocumentAPI.LineProperties()
lineProperties.style = DocumentAPI.LineStyle_Wave

borders = DocumentAPI.Borders()
borders = borders:setTop(lineProperties)
cell:setBorders(borders)
```

3.8.4.6 Таблица DocumentAPI.LineEndingStyle

В Таблице 53 приведены типы окончания линии. Используется в поле `style` таблицы `DocumentAPI.LineEndingProperties`.

Таблица 53 – Типы окончания линии

Наименование константы	Описание
DocumentAPI.LineEndingStyle_Arrow	
DocumentAPI.LineEndingStyle_Diamond	
DocumentAPI.LineEndingStyle_Oval	
DocumentAPI.LineEndingStyle_Stealth	
DocumentAPI.LineEndingStyle_Triangle	
DocumentAPI.LineEndingStyle_None	

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = tbl:getCell("C3")


lineProperties = DocumentAPI.LineProperties()
lineProperties.headLineEndingProperties = DocumentAPI.LineEndingProperties()
lineProperties.headLineEndingProperties.style =
DocumentAPI.LineEndingStyle_Oval


borders = DocumentAPI.Borders()
borders = borders:setTop(lineProperties)
cell:setBorders(borders)
```

3.8.5 Объединение и разделение ячеек таблицы

Допустимо объединение произвольного числа ячеек таблицы. При объединении указанный диапазон становится единой ячейкой. После завершения операции объединенная ячейка получает значение первой ячейки диапазона.

Для объединения нескольких ячеек используйте метод `CellRange.merge()`.

Пример:

```
-- Объединение ячеек A1 и A2 на первом листе табличного документа
local tbl = document:getBlocks():getTable(0)
tbl:getCellRange("A1:A2"):merge()
```

Допустимо разъединение только тех ячеек, которые были объединены ранее. После завершения операции данные, содержавшиеся в объединенной ячейке, будут помещены в верхнюю левую ячейку диапазона.

Для разъединения ячеек используйте метод `CellRange.unmerge()`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
-- Ячейка A1 является результатом объединения диапазона A1:A2
tbl:getCell("A1"):unmerge()
```

3.8.6 Таблица DocumentAPI.Table

Таблица `DocumentAPI.Table` предоставляет доступ к листу в табличном документе или таблице в составе текстового документа (см. Рисунок 1130).

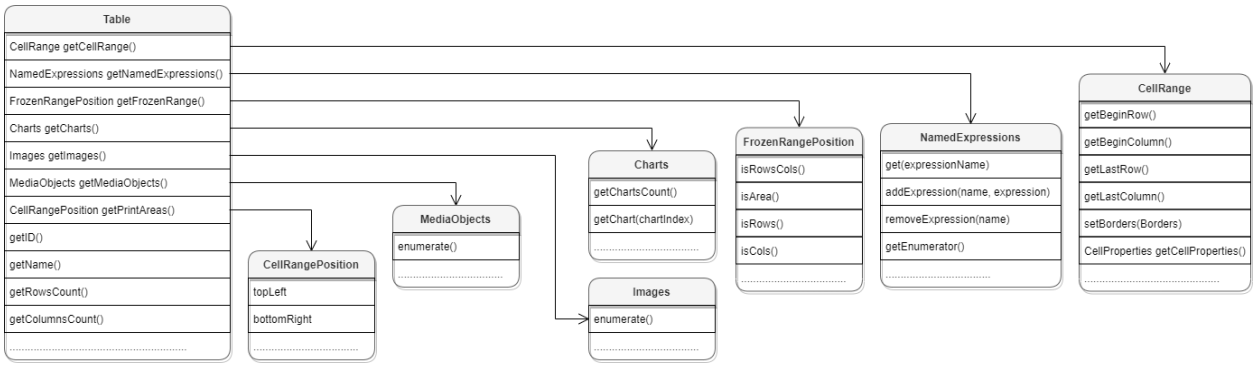


Рисунок 1130 – Структура полей таблицы `DocumentAPI.Table`

3.8.6.1 Метод Table:setName

Метод задает имя таблицы. В случае с табличным документом это имя будет являться заголовком листа документа. Данное значение должно быть уникальным, т.к. может использоваться для ссылки на таблицу, например, из формул.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:setName("Первый")
```

Для текстовых документов использование данного метода также допустимо, наименование таблицы нигде не отображается, но в дальнейшем его можно использовать для доступа к таблице по имени.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:setName("Первый")
tbl = document:getBlocks():getTable("Первый")
```

3.8.6.2 Метод Table:getName

Метод позволяет получить наименование листа табличного документа.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getName())
```

3.8.6.3 Метод Table:getRowsCount

Метод позволяет получить количество строк таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getRowsCount())
```

3.8.6.4 Метод Table:getColumnsCount

Метод позволяет получить количество столбцов таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getColumnsCount())
```

3.8.6.5 Метод Table:getCell

Метод позволяет получить доступ к отдельной ячейке таблицы. В качестве аргумента может выступать текстовое представление адреса ячейки, либо экземпляр таблицы DocumentApi.CellPosition.

Примеры:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
print(cell:getFormattedValue())

local cellPosition = DocumentAPI.CellPosition(2, 1)
local cell = tbl:getCell(cellPosition)
print(cell:getFormattedValue())
```

3.8.6.6 Метод Table:getCellRange

Метод позволяет получить доступ к диапазону ячеек таблицы DocumentAPI.CellRange.

Примеры:

```
local table = document:getBlocks():getTable(0)
local range = table:getCellRange("A1:C4")
for cell in range:enumerate() do
   print(cell:getFormattedValue())
end

local table = document:getBlocks():getTable(0)
local range = table:getCellRange(DocumentAPI.CellRangePosition(0, 0, 2, 2))
for cell in range:enumerate() do
   print(cell:getFormattedValue())
end
```

3.8.6.7 Метод Table:insertColumnAfter

Метод предназначен для вставки нового столбца после указанной позиции в таблице.

Вызов:

```
insertColumnAfter( columnIndex, copyColumnStyle, columnsCount )
```

Параметры:

- `columnIndex` – индекс столбца в таблице, после которого производится вставка. Индексация столбцов начинается с нуля.
- `copyColumnStyle` – флаг наследования стиля. Если этот параметр установлен в значение `true`, то новый столбец наследует настройки форматирования столбца с индексом `columnIndex`. Если параметр `copyColumnStyle` установлен в значение `false`, то настройки форматирования не копируются. Значение по умолчанию `true`.
- `columnsCount` – количество вставляемых столбцов. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")
-- Добавление двух столбцов в середину таблицы, без наследования настроек
форматирования
tbl:insertColumnAfter(0, false, 2)
```

3.8.6.8 Метод Table:insertColumnBefore

Метод предназначен для вставки нового столбца до указанной позиции в таблице.

Вызов:

```
insertColumnBefore( columnIndex, copyColumnStyle, columnsCount )
```

Параметры:

- `columnIndex` – индекс столбца в таблице, перед которым производится вставка. Индексация столбцов начинается с нуля.
- `copyColumnStyle` – флаг наследования стиля. Если этот параметр установлен в значение `true`, то новый столбец наследует настройки форматирования столбца с индексом `columnIndex`. Если параметр `copyColumnStyle` установлен в значение `false`, то настройки форматирования не копируются. Значение по умолчанию `true`.
- `columnsCount` – количество вставляемых столбцов. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")

-- Добавление двух столбцов в середину таблицы, без наследования настроек
форматирования
tbl:insertColumnBefore(1, false, 2)
```

3.8.6.9 Метод Table:insertRowAfter

Метод предназначен для вставки новой строки после указанной позиции в таблице.

Вызов:

```
insertRowAfter( rowIndex, copyRowStyle, rowsCount )
```

Параметры:

- `rowIndex` – индекс строки в таблице, после которой производится вставка. Индексация строк начинается с нуля.
- `copyRowStyle` – флаг наследования стиля. Если этот параметр установлен в значение `true`, то новая строка наследует настройки форматирования строки с индексом `rowIndex`. Если параметр `copyRowStyle` установлен в значение `false`, то настройки форматирования не копируются. Значение по умолчанию `true`.
- `rowsCount` – количество вставляемых строк. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")

-- Добавление двух строк в середину таблицы, без наследования настроек
форматирования
tbl:insertRowAfter(0, false, 2)
```

3.8.6.10 Метод Table:insertRowBefore

Метод предназначен для вставки новой строки до указанной позиции в таблице.

Вызов:

```
insertRowBefore( rowIndex, copyRowStyle, rowsCount )
```

Параметры:

- `rowIndex` – индекс строки в таблице, перед которой производится вставка. Индексация строк начинается с нуля.
- `copyRowStyle` – флаг наследования стиля. Если этот параметр установлен в значение `true`, то новая строка наследует настройки форматирования строки с индексом `rowIndex`. Если параметр `copyRowStyle` установлен в значение `false`, то настройки форматирования не копируются. Значение по умолчанию `true`.
- `rowsCount` – количество вставляемых строк. Значение по умолчанию 1.

Пример:

```
-- Создать в документе новую таблицу 2x2
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")


-- Добавление двух строк в середину таблицы, без наследования настроек
форматирования
tbl:insertRowBefore(1, false, 2)
```

3.8.6.11 Метод Table:removeColumn

Метод предназначен для удаления столбца таблицы, начиная с заданного индекса.

Вызов:

```
removeColumn(columnIndex, columnsCount)
```

Параметры:

- `columnIndex` – индекс столбца, начиная с которого будет удалено заданное количество столбцов. Индексация столбцов начинается с нуля.
- `columnsCount` – количество столбцов для удаления. Значение по умолчанию 1.

3.8.6.12 Метод Table:removeRow

Метод предназначен для удаления строки таблицы, начиная с заданного индекса.

Вызов:

```
removeRow(rowIndex, rowsCount)
```

Параметры:

- `rowIndex` – индекс строки, начиная с которого будет удалено `rowsCount` строк. Индексация строк начинается с нуля.
- `rowsCount` – количество строк для удаления. Значение по умолчанию 1.

3.8.6.13 Метод Table:setColumnWidth

Метод устанавливает ширину столбца таблицы в пунктах (1/72 дюйма).

Вызов:

```
setColumnWidth( columnIndex, width )
```

Параметры:

- `columnIndex` – индекс столбца в таблице, для которого устанавливается значение ширины. Индексация столбцов начинается с нуля.
- `width` – ширина столбца в пунктах (1/72 дюйма).

Пример:

```
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")


-- Установить ширину столбца в 400 pt
tbl:setColumnWidth(1,400)
```

3.8.6.14 Метод Table:setRowHeight

Метод устанавливает высоту строки таблицы в пунктах (1/72 дюйма).

Вызов:

```
setRowHeight(rowIndex, height)
```

Параметры:

- `rowIndex` – индекс строки в таблице, для которой устанавливается значение высоты. Индексация строк начинается с нуля.
- `height` – высота строки в пунктах (1/72 дюйма).
- `rowHeightRule` – точность значения (`DocumentAPI.RowHeightRule.Exact` – точно, `DocumentAPI.RowHeightRule.AtLeast` – не меньше).

Пример:

```
tbl = document:getRange():getBegin():insertTable(2, 2, "SomeTable")
```

```
-- Установить высоту строки в 100 pt
tbl:setRowHeight(1,100, DocumentAPI.RowHeightRule.Exact)
```

3.8.6.15 Метод Table:duplicate

Для создания копии листа в табличном документе используется метод `duplicate`. Созданная копия листа размещается после копируемого листа. Метод может быть использован только в табличном документе.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:duplicate()
```

3.8.6.16 Метод Table:remove

Для удаления таблицы в текстовом документе или листа в табличном документе используется метод `remove()`.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:remove()
```

3.8.6.17 Метод Table:moveTo

Для перемещения листа таблицы по указанному индексу в табличном документе используется метод `moveTo`. Указанный индекс должен быть меньше или равен количеству листов в документе. Индексация листов начинается с нуля. Метод может быть использован только в табличном документе.

Пример:

```
-- В табличном документе два листа с индексами 0 и 1.
-- Поменяем их местами.
local tbl = document:getBlocks():getTable(0)
tbl:moveTo(1)
```

3.8.6.18 Метод Table:setShowZeroValue

Для упрощения чтения таблицы нулевые значения ячеек могут быть скрыты. Для управления скрытием/показом ячеек используется метод `setShowZeroValue`.

Пример:

```
tbl = document:getBlocks():getTable(0)
tbl:setShowZeroValue(true)
```

3.8.6.19 Метод Table:getShowZeroValue

Для проверки режима отображения нулевых значений ячеек используется метод `getShowZeroValue`.

Пример:

```
tbl = document:getBlocks():getTable(0)
tbl:setShowZeroValue(false)
print(tbl:getShowZeroValue())
```

3.8.6.20 Метод Table:setVisible

Метод управляет видимостью листа таблицы. Используется только в табличном документе.

Вызов:

```
setVisible( visible )
```

Параметр:

`visible` – параметр, задающий видимость листа. Если значение параметра `visible` равно `true`, то лист таблицы отображается в редакторе таблиц.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:setVisible(false)
```

3.8.6.21 Метод Table:isVisible

Метод возвращает значение true, если лист таблицы в табличном документе отображается в редакторе таблиц.

Пример:

```
local tbl = document:getBlocks():getTable(0)
if not tbl:isVisible() then
  tbl:setVisible(true)
end
```

3.8.6.22 Метод Table:getFrozenRange

Существует возможность закрепления диапазона строк и столбцов. Такие диапазоны всегда остаются видимыми на экране в случае, когда пользователь осуществляет навигацию по таблице.

Метод `getFrozenRange` возвращает закрепленный диапазон `DocumentAPI.FrozenRangePosition`.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
local tbl = document:getBlocks():getTable(0)
tbl:freeze(frozenRangePosition)
print(tbl:getFrozenRange():isCols())
```

3.8.6.23 Метод Table:freeze

Метод `freeze` закрепляет заданную область DocumentAPI.FrozenRangePosition таблицы.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
local tbl = document:getBlocks():getTable(0)
tbl:freeze(frozenRangePosition)
print(tbl:getFrozenRange():isCols())
```

3.8.6.24 Метод Table:__eq

Метод используется для определения эквивалентности двух таблиц.

Пример:

```
local tbl1 = document:getBlocks():getTable(0)
local tbl2 = document:getBlocks():getTable("Table Name")
if tbl1 ~= nil and tbl2 ~= nil and tbl1 == tbl2 then
    print("tbl1 и tbl2 ссылаются на общую таблицу в документе")
end
```

3.8.6.25 Метод Table:__ne

Метод используется для определения неэквивалентности двух таблиц.

Пример:

```
local tbl1 = document:getBlocks():getTable(0)
local tbl2 = document:getBlocks():getTable("Table Name")
if tbl1 ~= nil and tbl2 ~= nil and tbl1 ~= tbl2 then
    print("tbl1 и tbl2 ссылаются на разные таблицы в документе")
end
```

3.8.6.26 Метод Table:setPrintArea

Метод служит для установки и сброса области печати DocumentAPI.CellRangePosition.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:setPrintArea(DocumentAPI.CellRangePosition(0, 0, 5, 5)) -- уставить
область печати размером в пять строк и пять колонок, начиная с левого
верхнего угла таблицы
```

3.8.6.27 Метод Table:setPrintAreas

Метод `Table:setPrintAreas` задает множественные области печати или экспорта `CellRangePositions`, где `CellRangePositions` - вектор из элементов `CellRangePosition` (см. описание вектора).

Пример:

```
tbl = document:getBlocks():getTable(0)
ranges = DocumentAPI.CellRangePositions()
ranges:push_back(DocumentAPI.CellRangePosition(0, 0, 5, 5))
ranges:push_back(DocumentAPI.CellRangePosition(1, 2, 5, 5))
tbl:setPrintAreas(ranges)


printAreas = tbl:getPrintAreas()
print(printAreas[0]:toString(), printAreas[1]:toString())
```

3.8.6.28 Метод Table:getPrintAreas

Метод `Table:getPrintAreas` возвращает текущие области печати - вектор элементов `DocumentAPI.CellRangePosition`. См. описание методов вектора.

Пример:

```
tbl = document:getBlocks():getTable(0)
tbl:setPrintArea(DocumentAPI.CellRangePosition(0, 0, 5, 5))


printAreas = tbl:getPrintAreas()
print(printAreas[0]:toString())
```

3.8.6.29 Метод Table:getCharts

Для получения списка диаграмм (`DocumentAPI.Charts`) таблицы используется метод `Table:getCharts`.

Пример:

```
for tbl in document:getBlocks():enumerateTables() do
  print(tbl:getCharts():getChartsCount())
end
```

3.8.6.30 Метод Table:getImages

Для получения списка изображений (DocumentAPI.Images) таблицы используется метод Table:getImages.

Пример:

```
tbl = document:getBlocks():getTable(0)
images = tbl:getImages()

for image in images:enumerate() do
  print(image)
end
```

3.8.6.31 Метод Table:getMediaObjects

Для получения списка медиаобъектов (DocumentAPI.MediaObjects) таблицы используется метод Table:getMediaObjects.

Пример:

```
tbl = document:getBlocks():getTable(0)
mediaObjects = tbl:getMediaObjects()

for mediaObject in mediaObjects:enumerate() do
  print(mediaObject)
end
```

3.8.6.32 Метод Table:getNamedExpressions

Для получения списка именованных выражений DocumentAPI:NamedExpressions используется метод Table:getNamedExpressions().

3.8.6.33 Группировка строк и колонок таблицы

Следующий набор методов позволяет группировать строки и колонки таблицы. Редактор дает возможность отображать группы в виде иерархии. Совместно с данными методами можно использовать методы Table::setColumnsVisible и Table::setRowsVisible чтобы раскрывать и закрывать фрагменты иерархии групп.

Методы могут вызвать исключения `OutOfRangeError` и `IncorrectArgumentError` в случае использования индексов, выходящих за рамки таблицы.

Методы для группировки:

```
groupRows(first, rowsCount)
ungroupRows(first, rowsCount)
clearRowGroups(first, rowsCount)
groupColumns(first, columnsCount)
ungroupColumns(first, columnsCount)
clearColumnnGroups(first, columnsCount)
```

3.8.6.34 Управление видимостью строк / колонок

Метод `Table::setColumnsVisible` позволяет задавать видимость `columnsCount` столбцов, начиная с индекса `first`. Индексация столбцов начинается с нуля.

```
setColumnsVisible(first, columnsCount, visible)
```

Метод `Table:setRowsVisible` позволяет задавать видимость `rowsCount` строк, начиная с индекса `first`. Индексация строк начинается с нуля.

```
setRowsVisible(first, rowsCount, visible)
```

3.8.7 Таблица DocumentAPI.Cell

Таблица `DocumentAPI.Cell` предоставляет доступ к ячейке в таблице текстового документа или на листе табличного документа (см. Рисунок 1131).

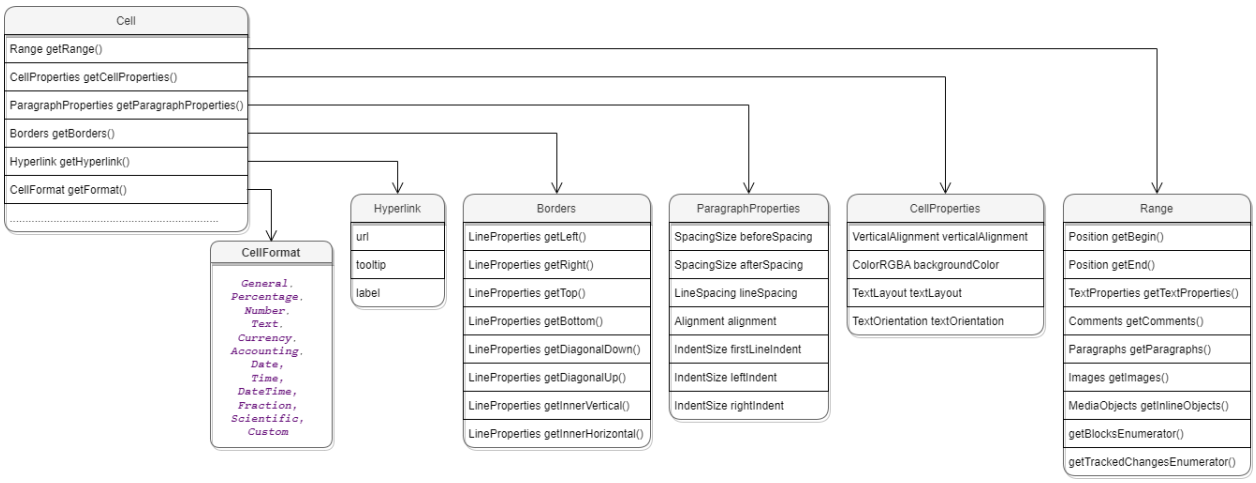


Рисунок 1131 – Объектная модель ячейки таблиц

3.8.7.1 Метод Cell:getRange

Метод возвращает объект DocumentAPI.Range для управления содержимым ячейки.

3.8.7.2 Метод Cell:setBorders

Метод предназначен для установки границ ячейки. Примеры использования приведены в разделе DocumentAPI.Borders.

3.8.7.3 Метод Cell:setFormula

Метод позволяет вставить формулу в ячейку табличного документа.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:getCell("A1"):setNumber(2.3)
tbl:getCell("A2"):setNumber(3.2)
tbl:getCell("A3"):setFormula("=SUM(A1:A2)") -- 5,5
```

3.8.7.4 Метод Cell:getFormat

Метод возвращает формат ячейки. Список поддерживаемых форматов ячеек приведен в разделе DocumentAPI.CellFormat.

Пример:

```
local table = document:getBlocks():getTable(0)
local cell = table:getCell("A1")


local cellFormatting = DocumentAPI.PercentageCellFormatting()
cell:setFormat(cellFormatting)
print("Формат: ", cell:getFormat()) -- 1
```

3.8.7.5 Метод Cell:setFormat

Метод устанавливает формат ячейки. Существуют несколько вариантов использования метода.

Варианты вызова метода:

```
setFormat(cellFormat)
```

Где **cellFormat** – формат ячейки типа DocumentAPI.CellFormat.

```
setFormat(accountingCellFormatting)
```

Где **accountingCellFormatting** – формат ячейки типа DocumentAPI.AccountingCellFormatting.

```
setFormat(percentageCellFormatting)
```

Где **percentageCellFormatting** – формат ячейки типа DocumentAPI.PercentageCellFormatting.

```
setFormat(numberCellFormatting)
```

Где **numberCellFormatting** – формат ячейки типа DocumentAPI.NumberCellFormatting.

```
setFormat(currencyCellFormatting)
```

Где **currencyCellFormatting** – формат ячейки типа DocumentAPI.CurrencyCellFormatting.

```
setFormat(dateTimeCellFormatting, typeFormat)
```

Где **dateTimeCellFormatting** – формат ячейки типа DocumentAPI.DateTimeCellFormatting, **typeFormat** - формат даты/времени типа DocumentAPI.CellFormat.

```
setFormat(fractionCellFormatting)
```

Где **fractionCellFormatting** – формат ячейки типа DocumentAPI.FractionCellFormatting.

```
setFormat(scientificCellFormatting)
```

Где **scientificCellFormatting** – формат ячейки типа DocumentAPI.ScientificCellFormatting.

Примеры использования:

```
local tbl = document:getBlocks():getTable(0)
local cellA1 = tbl:getCell("A1")


cellA1:setNumber(2.3)


-- Формат: Общий
cellA1:setFormat(DocumentAPI.CellFormat_General)
print("Формат Общий: ", cellA1:getFormat()) -- 0
```

```
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,3

-- Формат: Процентный
local a1CellFormatting = DocumentAPI.PercentageCellFormatting()
a1CellFormatting.decimalPlaces = 1
cellA1:setFormat(a1CellFormatting)
print("Формат Процентный: ", cellA1:getFormat()) -- 1
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 230,0%

-- Формат: Числовой
a1CellFormatting = DocumentAPI.NumberCellFormatting()
a1CellFormatting.decimalPlaces = 2
cellA1:setFormat(a1CellFormatting)
print("Формат Числовой: ", cellA1:getFormat()) -- 2
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30

-- Формат: Денежный
a1CellFormatting = DocumentAPI.CurrencyCellFormatting()
a1CellFormatting.symbol = '$'
cellA1:setFormat(a1CellFormatting)
print("Формат Денежный: ", cellA1:getFormat()) -- 4
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30$

-- Формат: Финансовый
a1CellFormatting = DocumentAPI.AccountingCellFormatting()
a1CellFormatting.symbol = '₽'
cellA1:setFormat(a1CellFormatting)
print("Формат Финансовый: ", cellA1:getFormat()) -- 5
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30₽

-- Формат: Дата / Время
a1CellFormatting = DocumentAPI.DateTimeCellFormatting()
a1CellFormatting.dateListID = DocumentAPI.DatePatterns_FullDate
a1CellFormatting.timeListID = DocumentAPI.TimePatterns_ShortTime
cellA1:setFormat(a1CellFormatting)
print("Формат Дата / Время: ", cellA1:getFormat()) -- 8
print("Значение ячейки: ", cellA1:getRange():extractText()) -- понедельник, 1
 января 1900 г. 7:12

-- Формат: Экспоненциальный
```

```
a1CellFormatting = DocumentAPI.FractionCellFormatting()
a1CellFormatting.minNumeratorDigits = 2
cellA1:setFormat(a1CellFormatting)
print("Формат Экспоненциальный: ", cellA1:getFormat()) -- 9
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2 2/7


-- Формат: Научный
a1CellFormatting = DocumentAPI.ScientificCellFormatting()
a1CellFormatting.decimalPlaces = 5
cellA1:setFormat(a1CellFormatting)
print("Формат Научный: ", cellA1:getFormat()) -- 10
print("Значение ячейки: ", cellA1:getRange():extractText()) -- 2,30000E+00
```

3.8.7.6 Метод Cell:getFormattedValue

Метод позволяет получить значение ячейки в текущем формате. Список поддерживаемых форматов см. в разделе DocumentAPI.CellFormat.

Пример:

```
local tbl = document:getBlocks():getTable(0)
print(tbl:getCell("A6"):getFormattedValue()) -- 21.6.1972
```

3.8.7.7 Метод Cell:setFormattedValue

Анализирует переданное значение и автоматически устанавливает формат ячейки и ее значение. В случае, если распознать тип переданного значения не удается, то для ячейки устанавливается формат DocumentAPI.CellFormat_Text.

Список поддерживаемых форматов см. в разделе DocumentAPI.CellFormat.

3.8.7.8 Метод Cell:unmerge

Разъединяет несколько ячеек, которые были объединены ранее.

Пример:

```
local tbl = document:getBlocks():getTable(0)
-- Ячейка A1 является результатом объединения диапазона A1:A2
tbl:getCell("A1"):unmerge()
```

3.8.7.9 Метод Cell:getHyperlink

Возвращает первый объект в ячейке типа DocumentAPI.Hyperlink.

```
local cell =
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))
local hyperlink = cell:getHyperlink()
```

3.8.7.10 Метод Cell:setContent

Определяет и устанавливает соответствующую формулу или значение, а затем форматирует ячейку. Устанавливает текст, если автоопределение не удалось.

Пример:

```
local cell =
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))
cell:setContent("=A2+A3")
```

3.8.7.11 Метод Cell:getBorders

Позволяет получить границы ячейки.

Пример:

```
local cell =
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))
local borders = cell:getBorders()
```

3.8.7.12 Метод Cell:getRawValue

Возвращает значение ячейки в формате «Общий» (без форматирования).

Пример:

```
local tbl = document:getBlocks():getTable(0)
local val = tbl:getCell("A6"):getRawValue()
```

3.8.7.13 Метод Cell:getCustomFormat

Возвращает строку формата ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cust_format = tbl:getCell("A6"):getCustomFormat()
```

3.8.7.14 Метод Cell:setCustomFormat

Устанавливает формат ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:getCell("A6"):setCustomFormat("0,00")
```

3.8.7.15 Метод Cell:setBool

Устанавливает для ячейки значение логического типа.

Пример:

```
local tbl = document:getBlocks():getTable(0)
tbl:getCell("A6"):setBool(true)
```

3.8.7.16 Метод Cell:setNumber

Устанавливает для ячейки значение числового типа.

Пример:

```
local sheet = document:getBlocks():getTable(0)
local cell = sheet:getCell(DocumentAPI.CellPosition(0, 0))
cell:setNumber(0.0001)
```

3.8.7.17 Метод Cell:setText

Устанавливает для ячейки значение строкового типа.

Пример:

```
local sheet = document:getBlocks():getTable(0)
local cell = sheet:getCell(DocumentAPI.CellPosition(0, 0))
local trackingChanges = "Disabled"
if document:isChangesTrackingEnabled() then
   trackingChanges = "Enabled"
end
cell:setText(trackingChanges)
```

3.8.7.18 Метод Cell:getFormulaAsString

Возвращает текст формулы ячейки. Формула – это любое выражение в ячейке, которое начинается со знака равенства (=).

Пример:

```
local tbl = document:getBlocks():getTable(0)
local formula = tbl:getCell("A6"):getFormulaAsString()
```

3.8.7.19 Метод Cell:getCellProperties

Позволяет получить свойства DocumentAPI.CellProperties ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell_props = tbl:getCell("A6"):getCellProperties()
```

3.8.7.20 Метод Cell:setCellProperties

Позволяет установить свойства ячейки DocumentAPI.CellProperties.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local props = tbl:getCell("A6"):getCellProperties()
props.verticalAlignment = DocumentAPI.VerticalAlignment_Center
tbl:getCell("A6"):setCellProperties(props)
```

3.8.7.21 Метод Cell:getParagraphProperties

Возвращает свойства абзаца DocumentAPI.ParagraphProperties, находящегося в ячейке.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A2") --(DocumentAPI.CellPosition(1,0))

local paraProps = cell:getParagraphProperties()
print(paraProps.alignment)
```

3.8.7.22 Метод Cell:setParagraphProperties

Устанавливает свойства абзаца DocumentAPI.ParagraphProperties, находящегося в ячейке.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A2") --(DocumentAPI.CellPosition(1,0))

local paraProps = cell:getParagraphProperties()
paraProps.alignment = DocumentAPI.Alignment_Center
cell:setParagraphProperties(paraProps)
```

3.8.7.23 Метод Cell:getPivotTable

Возвращает сводную таблицу DocumentAPI.PivotTable, относящуюся к ячейке.

Пример:

```
tbl = document:getBlocks():getTable(0)
cell = tbl:getCell("B4")
pivotTable = cell:getPivotTable()
```

3.8.8 Таблица DocumentAPI.CellProperties

Таблица `DocumentAPI.CellProperties` предназначена для форматирования содержимого в ячейках таблицы. Описание полей таблицы `DocumentAPI.CellProperties` представлено в Таблице 54.

Для задания свойств ячейки используется метод `Cell.setCellProperties()`. Для получения свойств ячейки используется метод `Cell.getCellProperties()`. Иерархия таблиц и полей `DocumentAPI.CellProperties` отображена на Рисунке 1132.

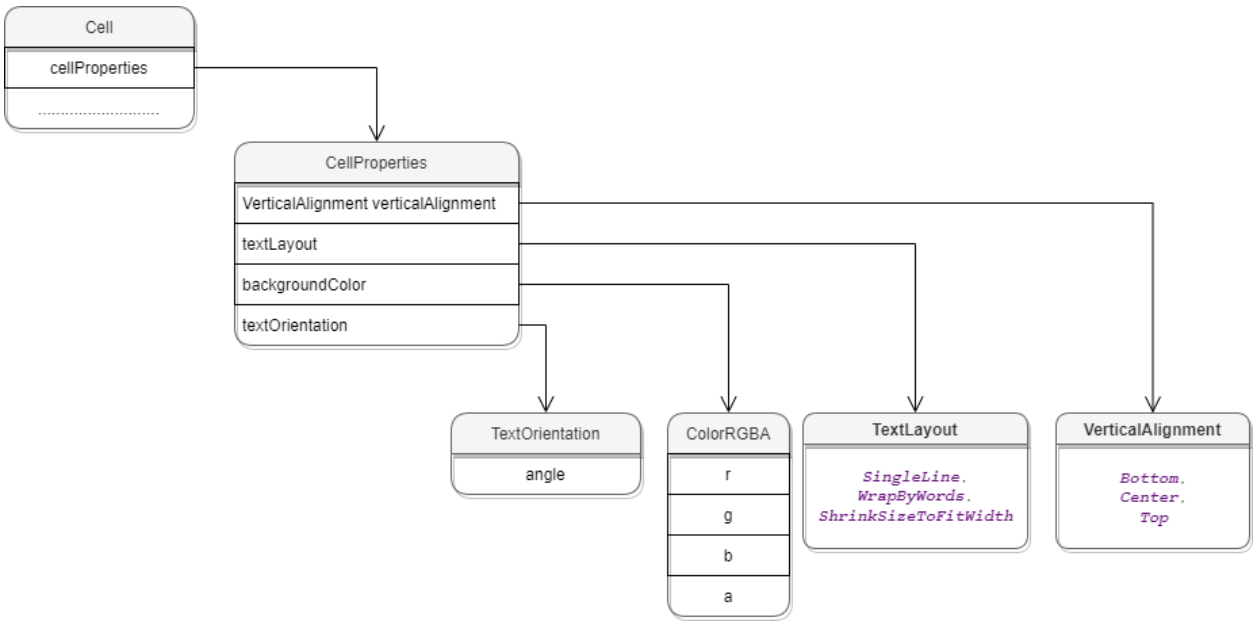


Рисунок 1132 – Объектная модель для работы со свойствами ячеек таблицы

Таблица 54 – Описание полей таблицы `DocumentAPI.CellProperties`

Поле	Тип	Значение
`CellProperties.verticalAlignment`	`VerticalAlignment`	Вертикальное выравнивание в ячейке.
`CellProperties.textLayout`	`TextLayout`	Способ отображения значения ячейки.
`CellProperties.backgroundColor`	`ColorRGBA`	Цвет фона ячейки.
`CellProperties.textOrientation`	`TextOrientation`	Ориентация текста в ячейке (угол поворота).

Пример:

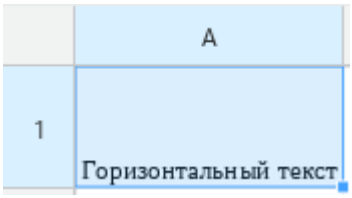
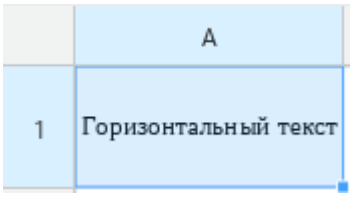
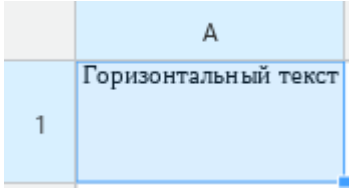
```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("D2") --(DocumentAPI.CellPosition(3,1))
```

```
local props = cell:getCellProperties()

props.verticalAlignment = DocumentAPI.VerticalAlignment_Center
props.textLayout = DocumentAPI.TextLayout_ShrinkSizeToFitWidth
props.backgroundColor = DocumentAPI.ColorRGBA(255, 255, 0, 255)
props.textOrientation = DocumentAPI.TextOrientation(45)

cell:setCellProperties(props)
```

3.8.9 Таблица DocumentAPI.VerticalAlignment

В Таблице 55 представлены константы видов выравнивания текста по вертикали. Используется в DocumentAPI.CellProperties, DocumentAPI.ShapeProperties.

Таблица 55 – Виды выравнивания текста по вертикали

Наименование константы	Представление в интерфейсе	
DocumentAPI.VerticalAlignment_Bottom		
DocumentAPI.VerticalAlignment_Center		
DocumentAPI.VerticalAlignment_Top		

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("A1")
local props = cell:getCellProperties()
props.verticalAlignment = DocumentAPI.VerticalAlignment_Center
cell:setCellProperties(props)
```

3.8.10 Таблица DocumentAPI.Hyperlink

Таблица `DocumentAPI.Hyperlink` описывает свойства ссылки. Может быть получена посредством вызова метода `Cell.getHyperlink()`.

Таблица 56 – Описание полей таблицы `DocumentAPI.Hyperlink`

Поле	Тип	Описание
`DocumentAPI.Hyperlink.url`	Строка	Адрес ссылки.
`DocumentAPI.Hyperlink.tooltip`	Строка	Текст подсказки.
`DocumentAPI.Hyperlink.label`	Строка	Текст описания.

Пример:

```
local cell =
document:getBlocks():getTable(0):getCell(DocumentAPI.CellPosition(0, 0))
local hyperlink = cell:getHyperlink()
print(hyperlink.url, hyperlink.tooltip, hyperlink.label)
```

3.8.10.1 Метод Hyperlink:__eq

Метод используется для определения эквивалентности двух объектов `Hyperlink`.

3.8.10.2 Метод Hyperlink:__ne

Метод используется для определения неэквивалентности двух объектов `Hyperlink`.

3.8.11 Таблица DocumentAPI.TextLayout

В Таблице 57 приведены варианты размещения текста в ячейках таблицы. Используется в методах `Cell.getCellProperties()`, `Cell.setCellProperties()`.

Таблица 57 – Варианты размещения текста в ячейках таблицы

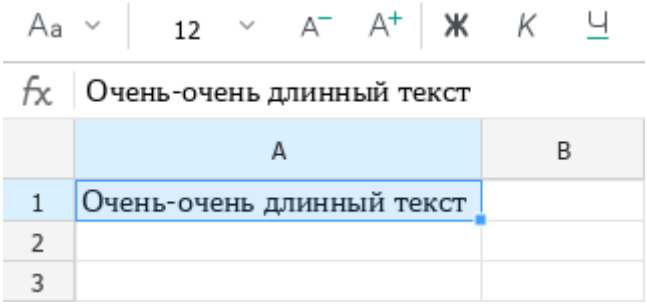
Наименование константы	Описание
`DocumentAPI.TextLayout_SingleLine`	Расположение текста в одну строку с наложением на соседние ячейки.
`DocumentAPI.TextLayout_WrapByWords`	Перенос данных внутри ячейки по

Наименование константы	Описание
	словам.
DocumentAPI.TextLayout_ShrinkSizeToFitWidth	Текст или число отображаются так, чтобы содержимое поместилось в ячейке полностью. Установленный размер шрифта не изменяется.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell_A1 = tbl:getCell("A1")
local props = cell_A1:getCellProperties()
props.textLayout = DocumentAPI.TextLayout_ShrinkSizeToFitWidth
cell_A1:setCellProperties(props)
```

Результат:



3.8.12 Таблица DocumentAPI.TextOrientation

Таблица DocumentAPI.TextOrientation предоставляет доступ к свойствам ориентации текста в ячейке, фигуре и т. д (см. DocumentAPI.CellProperties).

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("D2") --(DocumentAPI.CellPosition(3,1))
local props = cell:getCellProperties()
props.textOrientation = DocumentAPI.TextOrientation(45)
cell:setCellProperties(props)
print(props.textOrientation:getAngle())
```

3.8.12.1 Метод TextOrientation:getAngle

Возвращает угол ориентации текста в ячейке. Значение угла указывается в градусах.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("D2") --(DocumentAPI.CellPosition(3,1))
local cellProperties = cell:getCellProperties()
print(cellProperties.textOrientation:getAngle())
```

3.8.13 Таблица DocumentAPI.CellPosition

Таблица `DocumentAPI.CellPosition` позволяет задать координаты ячейки электронной таблицы или таблицы в составе текстового документа.

Позиция ячейки A1 имеет координаты (0, 0).

Также для указания адреса ячейки в качестве параметра метода `getCell` можно использовать строку вида «A1».

Примеры:

```
local table = document:getBlocks():getTable(0) -- первый лист документа
local cell = table:getCell(DocumentAPI.CellPosition(2, 0)) -- ячейка A3

local table = document:getBlocks():getTable(0) -- первый лист документа
local cell = table:getCell("A3") -- ячейка A3
```

3.8.13.1 Поле CellPosition.column

Номер столбца в значении ячейки. Нумерация столбцов начинается с нуля.

Пример:

```
cellPosition = DocumentAPI.CellPosition()
cellPosition.column = 1
```

3.8.13.2 Поле CellPosition.row

Номер строки в позиции ячейки. Нумерация строк начинается с нуля.

Пример:

```
cellPosition = DocumentAPI.CellPosition()
cellPosition.row = 1
```

3.8.13.3 Метод CellPosition:toString

Возвращает координаты ячейки в формате `(row: R, column: C)`, где `R` и `C` - номер строки и столбца соответственно.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local pos = DocumentAPI.CellPosition(0,0)
print(pos:toString()) --(row: 0, column: 0)
```

3.8.13.4 Метод CellPosition:__eq

Метод используется для определения эквивалентности двух объектов `CellPosition`.

3.8.13.5 Метод CellPosition:__ne

Метод используется для определения неэквивалентности двух объектов `CellPosition`.

3.8.14 Таблица DocumentAPI.CellRange

Таблица `DocumentAPI.CellRange` описывает диапазон ячеек таблицы.

Пример:

```
local cellRange = table:getCellRange("B3:C4")
```

3.8.14.1 Метод CellRange:enumerate

Метод возвращает коллекцию ячеек в диапазоне.

Пример:

```
-- Печать значений ячеек в диапазоне B3:C4
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
for cell in rng:enumerate() do
    print(cell:getFormattedValue())
end
```

3.8.14.2 Метод CellRange:getBeginRow

Метод возвращает индекс строки первой ячейки диапазона. Нумерация строк начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getBeginRow()) -- 2
```

3.8.14.3 Метод CellRange:getBeginColumn

Метод возвращает индекс столбца первой ячейки диапазона. Нумерация столбцов начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getBeginColumn()) -- 1
```

3.8.14.4 Метод CellRange:getLastRow

Метод возвращает индекс строки последней ячейки диапазона. Нумерация строк начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getLastRow()) -- 3
```

3.8.14.5 Метод CellRange:getLastColumn

Метод возвращает индекс столбца последней ячейки диапазона. Нумерация столбцов начинается с нуля.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
print(rng:getLastColumn()) -- 2
```

3.8.14.6 Метод CellRange:setBorders

Метод предназначен для установки границ диапазона ячеек. Отдельные границы устанавливаются с помощью методов таблицы DocumentAPI.Borders.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("C3")
--
line_prop = DocumentAPI.LineProperties()
line_prop.style = DocumentAPI.LineStyle_Dash
line_prop.width = 1.5
line_prop.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0, 255))
--
newBorders = DocumentAPI.Borders()
```

```
newBorders = newBorders:setLeft(line_prop)
newBorders = newBorders:setRight(line_prop)
newBorders = newBorders:setTop(line_prop)
newBorders = newBorders:setBottom(line_prop)
--
local borders = cell:setBorders(newBorders)
```

3.8.14.7 Метод CellRange:insertCurrentDateTime

Метод служит для установки значения даты/времени DocumentAPI.DateTimeFormat диапазона ячеек.

3.8.14.8 Метод CellRange:getCellProperties

Метод возвращает набор свойств форматирования (DocumentAPI.CellProperties) для диапазона ячеек. Возвращаемая структура содержит свойства, общие для всех ячеек диапазона.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
local cellProperties = rng:getCellProperties()
print(cellProperties.backgroundColor.r)
```

3.8.14.9 Метод CellRange:setCellProperties

Метод предназначен для установки свойств DocumentAPI.CellProperties для всех ячеек диапазона.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local rng = tbl:getCellRange("B3:C4")
---
local props = DocumentAPI.CellProperties()
props.backgroundColor = DocumentAPI.ColorRGBA(55, 146, 179, 200)
rng:setCellProperties(props)
```

3.8.14.10 Метод CellRange:merge

Метод объединяет несколько ячеек таблицы в одну. Группа ячеек (диапазон) формируется с помощью таблицы `CellRange`. Содержимое крайней левой ячейки диапазона помещается в объединенной ячейке.

Пример:

```
-- Объединение ячеек A1 и A2 на первом листе табличного документа
local tbl = document:getBlocks():getTable(0)
tbl:getCellRange("A1:A2"):merge()
```

3.8.14.11 Метод CellRange:unmerge

Метод разъединяет ранее объединенные ячейки.

Пример:

```
local tbl = document:getBlocks():getTable(0)
-- Ячейка A1 является результатом объединения диапазона A1:A2
tbl:getCell("A1"):unmerge()
```

3.8.15 Таблица DocumentApi.DateTimeFormat

В Таблице 58 представлены варианты масштабирования при печати табличных документов. Используется в качестве параметра метода `CellRange.insertCurrentDateTime()`.

Таблица 58 – Варианты масштабирования при печати табличных документов

Наименование константы	Описание
`DocumentAPI.DateTimeFormat_DateTime`	Дата/время.
`DocumentAPI.DateTimeFormat_Date`	Дата.
`DocumentAPI.DateTimeFormat_Time`	Время.

3.8.16 Таблица DocumentAPI.CellRangePosition

Таблица `DocumentAPI.CellRangePosition` представляет положение диапазона ячеек в таблице. Используется в качестве поля `tableRange` таблицы `DocumentAPI.TableRangeInfo`, а также в методах `Table.getCellRange()`, `Chart.setRange()` По умолчанию диапазон включает одну ячейку в позиции 0,0 что соответствует верхней левой ячейке таблицы для редактора текста, либо ячейке A1 для редактора таблиц.

Описание полей таблицы `DocumentAPI.CellRangePosition` представлено в Таблице 59.

Таблица 59 – Поля таблицы `DocumentAPI.CellRangePosition`

Поле	Тип	Описание
`topLeft`	CellPosition	Позиция левой верхней ячейки таблицы прямоугольного диапазона. Значение 0,0 соответствует верхней левой ячейке таблицы для редактора текста, либо ячейке A1 для редактора таблиц.
`bottomRight`	CellPosition	Содержит позицию правой нижней ячейки таблицы прямоугольного диапазона.

Примеры:

```
local table = document:getBlocks():getTable(0)
local cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
local range = table:getCellRange(cellRangePosition)

local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)
local cellRangePosition = rangeInfo.tableRangeInfo.tableRange
print("topLeft=", cellRangePosition.topLeft.row,
cellRangePosition.topLeft.column)
print("bottomRight=", cellRangePosition.bottomRight.row,
cellRangePosition.bottomRight.column)
```

3.8.16.1 Метод CellRangePosition:toString

Возвращает информацию о диапазоне ячеек в виде строкового значения формата `(topLeft: <value>, bottomRight: <value>)`.

Пример:

```
local cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
print(cellRangePosition:toString()) -- [topLeft: (row: 0, column: 0), bottomR
ight: (row: 5, column: 5)]
```

3.8.16.2 Метод CellRangePosition:__eq

Метод используется для определения эквивалентности двух объектов `CellRangePosition`.

3.8.16.3 Метод CellRangePosition:__ne

Метод используется для определения неэквивалентности двух объектов `CellRangePosition`.

3.8.17 Таблица DocumentAPI.FrozenRangePosition

Таблица `DocumentAPI.FrozenRangePosition` представляет заблокированную область таблицы. Возвращается посредством метода `Table.getFrozenRange()`, устанавливается методом `Table.freeze()`.

3.8.17.1 Конструкторы

Конструктор с параметрами по умолчанию.

```
FrozenRangePosition()
```

Конструктор, создающий диапазон ячеек. В качестве параметров используются координаты левой верхней и правой нижней точек области.

```
FrozenRangePosition(top, left, bottom, right)
```

Примеры:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition()
print(frozenRangePosition:isRowsCols())


frozenRangePosition = DocumentAPI.FrozenRangePosition(0, 2, 5, 5)
print(frozenRangePosition:isRowsCols())
```

3.8.17.2 Метод FrozenRangePosition:create

Создает объект заблокированной области таблицы `FrozenRangePosition`. В качестве параметров используются координаты левой верхней и правой нижней точек области.

Вызов:

```
FrozenRangePosition create(top, left, bottom, right)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.create(0, 2, 5, 5)
print(frozenRangePosition:isRowsCols())
```

3.8.17.3 Метод FrozenRangePosition:createFrozenArea

Создает объект заблокированной области таблицы `FrozenRangePosition`. Область содержит все ячейки прямоугольника `{0, 0, bottom, right}`.

Вызов:

```
FrozenRangePosition createFrozenArea(bottom, right)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenArea(0, 2)
print(frozenRangePosition:isArea())
```

3.8.17.4 Метод FrozenRangePosition:createFrozenRows

Создает объект заблокированной области таблицы `FrozenRangePosition`. Область содержит все строки с `first` по `last`.

Вызов:

```
FrozenRangePosition createFrozenRows(first, last)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenRows(0, 2)
print(frozenRangePosition:isRows())
```

3.8.17.5 Метод FrozenRangePosition:createFrozenCols

Создает объект заблокированной области таблицы `FrozenRangePosition`. Область содержит все колонки с `first` по `last`.

Вызов:

```
FrozenRangePosition createFrozenCols(first, last)
```

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
print(frozenRangePosition:isRowsCols())
```

3.8.17.6 Метод FrozenRangePosition:isRowsCols

Возвращает `true` если диапазон содержит строки и колонки.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenArea(2, 2)
print(frozenRangePosition:isRowsCols())
```

3.8.17.7 Метод FrozenRangePosition:isArea

Возвращает `true` если диапазон является непрерывной областью.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenArea(2, 2)
print(frozenRangePosition:isArea())
```

3.8.17.8 Метод FrozenRangePosition:isRows

Возвращает `true` если диапазон состоит из строк.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenRows(0, 2)
print(frozenRangePosition:isRows())
```

3.8.17.9 Метод FrozenRangePosition:isCols

Возвращает `true` если диапазон состоит из колонок.

Пример:

```
frozenRangePosition = DocumentAPI.FrozenRangePosition.createFrozenCols(0, 2)
print(frozenRangePosition:isCols())
```

3.8.17.10 Метод FrozenRangePosition:__eq

Метод используется для определения эквивалентности двух объектов `FrozenRangePosition`.

3.8.17.11 Метод FrozenRangePosition:__ne

Метод используется для определения неэквивалентности двух объектов `FrozenRangePosition`.

3.8.18 Таблица DocumentAPI.TableRangeInfo

Таблица `DocumentAPI.TableRangeInfo` описывает диапазон ячеек таблицы.

Описание полей таблицы `DocumentAPI.TableRangeInfo` представлено в Таблице 60.

Таблица 60 – Поля таблицы `DocumentAPI.TableRangeInfo`

Поле	Тип	Описание
`tableRange`	`DocumentAPI.CellRangePosition`	Диапазон ячеек.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local charts = tbl:getCharts()
local rangeInfo = charts:getChart(0):getRange(0)
local tableRangeInfo = rangeInfo.tableRangeInfo
local tableRange = tableRangeInfo.tableRange
print("topLeft=", tableRange.topLeft.row, tableRange.topLeft.column)
print("topLeft=", tableRange.bottomRight.row, tableRange.bottomRight.column)
```

3.9 Сводные таблицы

Сводная таблица - инструмент обработки данных, служащий для их обобщения и удобства обработки. Схема взаимодействия объектов, связанных со сводными таблицами, приведена на Рисунке 1133.

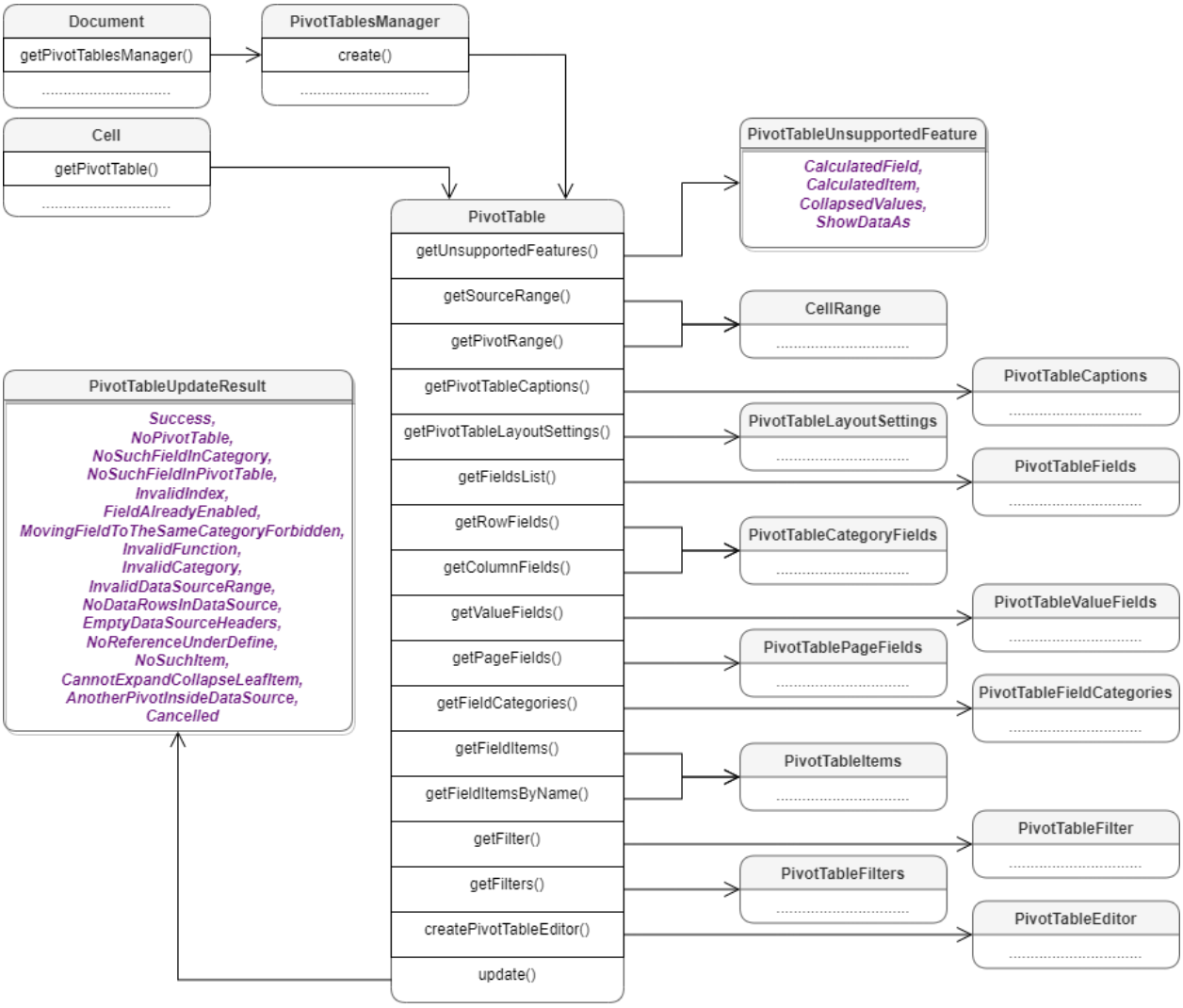


Рисунок 1133 – Сводные таблицы

3.9.1 Таблица DocumentAPI.PivotTablesManager

Таблица `PivotTablesManager` используется для создания сводных таблиц, содержит метод `create()`. Может быть получена вызовом `Document.getPivotTablesManager()`.

Пример:

```
local pivotTablesManager = document:getPivotTablesManager()
```

3.9.1.1 Метод PivotTablesManager:create

Метод создает сводную таблицу PivotTable на основе диапазона исходных данных CellRange.

Если местоположение не задано, создается новый лист (таблица), и сводная таблица будет расположена по умолчанию.

Пример:

```
local pivotTablesManager = document:getPivotTablesManager()
local tbl = document:getBlocks():getTable(0)
local cellRange = tbl:getCellRange("I3:K7")
local pivotTable = pivotTablesManager:create(cellRange, tbl:getCell("L8"))
```

3.9.2 Таблица DocumentAPI.PivotTable

Таблица для представления сводной таблицы. Может быть получена из ячейки Cell.getPivotTable(), либо получена при создании новой сводной таблицы PivotTablesManager.create().

3.9.2.1 Метод PivotTable:remove

Метод удаляет сводную таблицу.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
pivotTable:remove()
```

3.9.2.2 Метод PivotTable:getSourceRangeAddress

Метод возвращает текстовое представление диапазона исходных данных сводной таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
print(pivotTable:getSourceRangeAddress()) -- 'Sheet1'!I3:K7
```

3.9.2.3 Метод PivotTable:getSourceRange

Метод возвращает диапазон DocumentAPI.CellRange исходных данных сводной таблицы.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
local cellRange = pivotTable:getSourceRange()
print(cellRange:getBeginRow(), cellRange:getLastRow()) -- 2  6
```

3.9.2.4 Метод PivotTable:getPivotRange

Метод возвращает диапазон ячеек DocumentAPI.CellRange, в котором размещена сводная таблица.

Пример:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
local cellRange = pivotTable:getPivotRange()
print(cellRange:getBeginRow(), cellRange:getLastRow()) -- 7  10
```

3.9.2.5 Метод PivotTable:changeSourceRange

Метод позволяет задать новый диапазон исходных данных сводной таблицы без обновления самой таблицы. Параметр `sourceRange` – строка, представляющая новый диапазон таблицы.

Пример:

```
pivotTable:changeSourceRange("I3:K5")
local cellRange = pivotTable:getSourceRange()
print(cellRange:getBeginRow(), cellRange:getLastRow())
```

3.9.2.6 Метод PivotTable:isRowGrandTotalEnabled

Метод возвращает `true`, если разрешено показывать общие итоги для строк.

3.9.2.7 Метод PivotTable:isColumnGrandTotalEnabled

Метод возвращает `true`, если разрешено показывать общие итоги для столбцов.

3.9.2.8 Метод PivotTable:getPivotTableCaptions

Метод возвращает информацию DocumentAPI.PivotTableCaptions о всех заголовках сводной таблицы.

Пример:

```
local pivotTableCaptions = pivotTable:getPivotTableCaptions()
print(pivotTableCaptions.grandTotalCaption)
print(pivotTableCaptions.valuesHeaderCaption)
print(pivotTableCaptions.rowHeaderCaption)
print(pivotTableCaptions.columnHeaderCaption)
print(pivotTableCaptions.errorCaption)
print(pivotTableCaptions.emptyCaption)
```

3.9.2.9 Метод PivotTable:getPivotTableLayoutSettings

Метод возвращает настройки отображения DocumentAPI.PivotTableLayoutSettings сводной таблицы.

Пример:

```
local settings = pivotTable:getPivotTableLayoutSettings()
print(settings.reportLayout)
print(settings.valueFieldsOrientation)
print(settings.pageFieldOrder)
print(settings.indentForCompactLayout)
print(settings.pageFieldWrapCount)
```

3.9.2.10 Метод PivotTable:getUnsupportedFeatures

Метод возвращает неподдерживаемые свойства DocumentAPI.PivotTableUnsupportedFeature сводной таблицы.

Пример:

```
local unsupportedFeatures = pivotTable:getUnsupportedFeatures()
for featureIndex = 0, unsupportedFeatures:size() - 1 do
   print(unsupportedFeatures[featureIndex])
end
```

3.9.2.11 Метод PivotTable:getFieldsList

Метод возвращает список DocumentAPI.PivotTableField всех полей сводной таблицы.

Пример:

```
local fieldsList = pivotTable:getFieldsList()
print(fieldsList:size())
for fieldIdx = 0, fieldsList:size() - 1 do
   print(fieldsList[fieldIdx].fieldProperties.fieldName)
end
```

3.9.2.12 Метод PivotTable:getRowFields

Метод возвращает список полей DocumentAPI.PivotTableCategoryField из области строк.

Пример:

```
local rowFields = pivotTable:getRowFields()
for fieldIdx = 0, rowFields:size() - 1 do
   print(rowFields[fieldIdx].fieldProperties.fieldName)
end
```

3.9.2.13 Метод PivotTable:getColumnFields

Метод возвращает список полей DocumentAPI.PivotTableCategoryField из области колонок.

Пример:

```
local columnFields = pivotTable:getColumnFields()
for fieldIdx = 0, columnFields:size() - 1 do
   print(columnFields[fieldIdx].fieldProperties.fieldName)
end
```

3.9.2.14 Метод PivotTable:getValueFields

Метод возвращает список полей DocumentAPI.PivotTableValueField из области значений.

Пример:

```
local valueFields = pivotTable:getValueFields()
for fieldIdx = 0, valueFields:size() - 1 do
   print(valueFields[fieldIdx].baseFieldName)
   print(valueFields[fieldIdx].valueFieldName)
   print(valueFields[fieldIdx].cellNumberFormat)
   print(valueFields[fieldIdx].totalFunction)
end
```

3.9.2.15 Метод PivotTable:getPageFields

Метод возвращает список полей DocumentAPI.PivotTablePageField из области фильтров.

Пример:

```
local pageFields = pivotTable:getPageFields()
print(pageFields:size())
```

3.9.2.16 Метод PivotTable:getFieldCategories

Метод возвращает список категорий DocumentAPI.PivotTableFieldCategories, содержащих заданное поле fieldName.

Пример:

```
local fieldCategories = pivotTable:getFieldCategories("Age")
```

3.9.2.17 Метод PivotTable:getFieldItems

Метод возвращает все элементы DocumentAPI.PivotTableItems сводной таблицы по заданному имени поля fieldName.

Пример:

```
local pivotTableItems = pivotTable:getFieldItems()
print(pivotTableItems)
```

3.9.2.18 Метод PivotTable:getFieldItemsByName

Метод возвращает все элементы DocumentAPI.PivotTableItems из заданного поля fieldName по имени itemName.

Пример:

```
local pivotTableItemsByName = pivotTable:getFieldItemsByName("Ultimate Questi
on of Life", "42")
print(pivotTableItemsByName)
```

3.9.2.19 Метод PivotTable:getFilter

Метод возвращает фильтр DocumentAPI.PivotTableFilter по заданному имени поля fieldName.

Пример:

```
local filter = pivotTable:getFilter("Age")
print(filter:getFieldName())
```

3.9.2.20 Метод PivotTable:getFilters

Метод возвращает список фильтров DocumentAPI.PivotTableFilter сводной таблицы.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   -- use filter
end
```

3.9.2.21 Метод PivotTable:update

Метод обновляет и полностью пересчитывает сводную таблицу, возвращает DocumentAPI.PivotTableUpdateResult.

Пример:

```
local updateResult = pivotTable:update()
if (updateResult ~= DocumentAPI.PivotTableUpdateResult_Success) then
   print(updateResult)
end
```

3.9.2.22 Метод PivotTable:createPivotTableEditor

Метод возвращает объект DocumentAPI.PivotTableEditor, который служит для обновления свойств и редактирования сводной таблицы.

Пример:

```
local cell = tbl:getCell("L8")
local pivotTable = cell:getPivotTable()
local pivotTableEditor = pivotTable:createPivotTableEditor()
```

3.9.3 Таблица DocumentAPI.PivotTableCaptions

Таблица DocumentAPI.PivotTableCaptions хранит все пользовательские заголовки сводной таблицы. Описание полей таблицы представлено в Таблице 61.

Таблица 61 – Описание полей таблицы DocumentAPI.PivotTableCaptions

Поле	Описание
PivotTableCaptions.errorCaption	Алиас для значений, которые возвращают ошибку.
PivotTableCaptions.emptyCaption	Алиас для значений, которые возвращают пустое значение.
PivotTableCaptions.grandTotalCaption	Алиас общих итогов.
PivotTableCaptions.valuesHeaderCaption	Алиас поля из области значений; это поле отображается в отчете в случае, если в сводной таблице наличие более двух полей из области значений, и макет имеют тип ‘outline’ или ‘tabular’.
PivotTableCaptions.rowHeaderCaption	Алиас заголовка строк (виден только при включенном компактном макете, это алиас по умолчанию).
PivotTableCaptions.columnHeaderCaption	Алиас заголовка колонок (виден

Поле	Описание
	только при включенном компактном макете, это алиас по умолчанию).

3.9.4 Таблица DocumentAPI.PivotTableLayoutSettings

Таблица `DocumentAPI.PivotTableLayoutSettings` содержит настройки отображения сводной таблицы. Данная таблица может быть получена в результате вызова `PivotTable.getPivotTableLayoutSettings()` и установлена методом `PivotTableEditor.setLayoutSettings()`. Описание полей таблицы представлено в Таблице 62.

Таблица 62 – Описание полей таблицы `DocumentAPI.PivotTableLayoutSettings`

Поле	Описание
`PivotTableLayoutSettings.reportLayout`	Настройка вида макета сводной таблицы (`PivotTableReportLayout`: компактный, табличный, структурный).
`PivotTableLayoutSettings.valueFieldsOrientation`	Настраивает положение значений в случае, если в сводной таблице более двух полей значений. Тип - `ValueFieldsOrientation`.
`PivotTableLayoutSettings.pageFieldOrder`	Настройка порядка полей фильтров (`PageFieldOrder`: вниз, затем поперек или сначала поперек, потом вниз).
`PivotTableLayoutSettings.indentForCompactLayout`	Размер отступа для полей в области строк в компактном макете (режим иерархии в случае наличия более двух полей).
`PivotTableLayoutSettings.pageFieldWrapCount`	Настройка связана с `pageFieldOrder`, она показывает через сколько полей будет совершено указанное действие (перенос на следующую строку и т.д).
`PivotTableLayoutSettings.isMergeAndCenterLabelsEnabled`	Настройка позволяет объединить ячейки заголовков.
`PivotTableLayoutSettings.useGridDropZones`	Флаг, отвечающий за отображение классического вида (как в Excel 2003). Влияет только на расположение полей в отчете.
`PivotTableLayoutSettings.displayFieldCaptions`	Флаг, отвечающий за отображение заголовков полей.

3.9.5 Таблица DocumentAPI.PivotTableReportLayout

Таблица `DocumentAPI.PivotTableReportLayout` описывает внешний вид отчетов сводной таблицы. Является полем таблицы

DocumentAPI.PivotTableLayoutSettings. Описание полей таблицы представлено в Таблице 63.

Таблица 63 – Описание полей таблицы DocumentAPI.PivotTableReportLayout

Поле	Описание
DocumentAPI.PivotTableReportLayout_Compact	Компактный вид.
DocumentAPI.PivotTableReportLayout_Tabular	Табличный вид.
DocumentAPI.PivotTableReportLayout_Outline	Структурный вид.

3.9.6 Таблица DocumentAPI.ValueFieldsOrientation

Таблица DocumentAPI.ValueFueldsOrientation описывает варианты ориентации в случае, когда в сводной таблице более, чем одно поле из области значений. Является полем таблицы DocumentAPI.PivotTableLayoutSettings. Описание полей таблицы представлено в Таблице 64.

Таблица 64 – Описание полей таблицы DocumentAPI.ValueFueldsOrientation

Поле	Описание
DocumentAPI.ValueFueldsOrientation_ByRows	По строкам.
DocumentAPI.ValueFueldsOrientation_ByColumns	По столбцам.

3.9.7 Таблица DocumentAPI.PageFieldOrder

Таблица DocumentAPI.PageFieldOrder описывает вид отображения полей из области фильтров. Является полем таблицы DocumentAPI.PivotTableLayoutSettings. Описание полей таблицы представлено в Таблице 65.

Таблица 65 – Описание полей таблицы DocumentAPI.PageFieldOrder

Поле	Описание
DocumentAPI.PageFieldOrder_DownThenOver	Вниз, затем поперек.
DocumentAPI.PageFieldOrder_OverThenDown	Поперек, затем вниз.

3.9.8 Таблица DocumentAPI.PivotTableUnsupportedFeature

Таблица DocumentAPI.PivotTableUnsupportedFeature описывает неподдерживаемую функциональность сводных таблиц. Получение неподдерживаемой функциональности сводных таблиц описано в PivotTable:getUnsupportedFeatures(). Описание полей таблицы представлено в Таблице 66.

Таблица 66 – Описание полей таблицы DocumentAPI.PivotTableUnsupportedFeature

Поле	Описание
DocumentAPI.PivotTableUnsupportedFeature_CalculatedField	Вычисляемые поля.
DocumentAPI.PivotTableUnsupportedFeature_CalculatedItem	Вычисляемые элементы.
DocumentAPI.PivotTableUnsupportedFeature_CollapsedValues	Свернутые поля.
DocumentAPI.PivotTableUnsupportedFeature_ShowDataAs	Вычисления (Show data как в MS Excel).

3.9.9 Таблица DocumentAPI.PivotTableFieldCategories

Класс обеспечивает доступ к списку категорий поля сводной таблицы. Может быть получена посредством использования метода PivotTable.getFieldCategories().

3.9.9.1 Метод PivotTableFieldCategories:enumerate

Метод для перечисления категорий поля DocumentAPI.PivotTableFieldCategory.

Пример:

```
local fieldCategories = pivotTable:getFieldCategories("Age")
for fieldCategory in fieldCategories:enumerate() do
   print(fieldCategory)
end
```

3.9.10 Таблица DocumentAPI.PivotTableFunction

Таблица DocumentAPI.PivotTableFunction описывает функции, которые могут быть использованы в сводных таблицах. Описание полей таблицы представлено в Таблице 67. Таблица используется в качестве поля subtotalFunctions таблицы DocumentAPI.PivotTableCategoryField.

Таблица 67 – Описание полей таблицы DocumentAPI.PivotTableFunction

Поле	Описание
DocumentAPI.PivotTableFunction_Auto	Автозаполнение.
DocumentAPI.PivotTableFunction_Sum	Суммирует все числовые данные.
DocumentAPI.PivotTableFunction_Count	Количество всех ячеек.
DocumentAPI.PivotTableFunction_CountNums	Количество числовых ячеек.

Поле	Описание
DocumentAPI.PivotTableFunction_Average	Среднее значение.
DocumentAPI.PivotTableFunction_Max	Наибольшее значение.
DocumentAPI.PivotTableFunction_Min	Наименьшее значение.
DocumentAPI.PivotTableFunction_Product	Произведение всех ячеек.
DocumentAPI.PivotTableFunction_StdDeviation	Стандартное смещенное отклонение.
DocumentAPI.PivotTableFunction_StdDeviationPopulation	Стандартное несмещенное отклонение.
DocumentAPI.PivotTableFunction_Variance	Смещенная дисперсия.
DocumentAPI.PivotTableFunction_VariancePopulation	Несмещенная дисперсия.

3.9.11 Таблица DocumentAPI.PivotTableFilters

Таблица обеспечивает доступ к списку фильтров. Для получения DocumentAPI.PivotTableFilters используется метод PivotTable.getFilters().

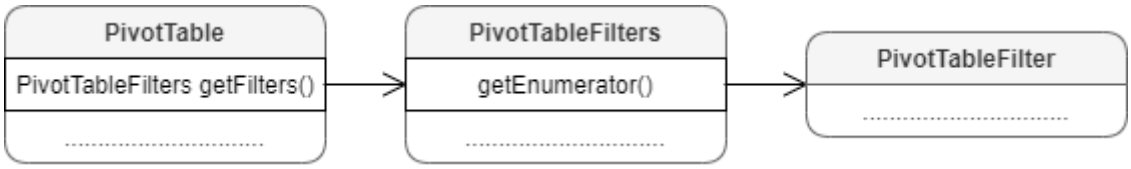


Рисунок 1134 – Объектная модель таблиц для работы с фильтрами

3.9.11.1 Метод PivotTableFilters:enumerate

Метод используется для доступа к коллекции фильтров (см. DocumentAPI.PivotTableFilter).

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   print(filter:getName(0))
   print(filter:getFieldName())
end
```

3.9.12 Таблица DocumentAPI.PivotTableFilter

Позволяет осуществить доступ к списку фильтров таблицы, каждый из которых обладает свойством видимости. При любом изменении фильтров они должны быть применены к сводной таблице посредством использования методов PivotTableEditor.setFilter(), PivotTableEditor.setFilters().

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   for filterIdx = 0, filter:getCount() - 1 do
      filter:setHidden(filterIdx, false)
   end
end
pivotTableEditor:setFilters(filters)
pivotTableEditor:apply()
```

3.9.12.1 Метод PivotTableFilter:getFieldName

Возвращает имя поля, с которым ассоциирован фильтр.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   for filterIdx = 0, filter:getCount() - 1 do
      print(filter:getFieldName())
   end
end
```

3.9.12.2 Метод PivotTableFilter:getCount

Возвращает количество фильтруемых полей.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   print(filter:getCount())
end
```

3.9.12.3 Метод PivotTableFilter:getName

Возвращает имя поля для заданного индекса.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   for filterIdx = 0, filter:getCount() - 1 do
      print(filter:getName(filterIdx))
   end
end
```

3.9.12.4 Метод PivotTableFilter:isHidden

Возвращает видимость поля для заданного индекса `itemIndex`. Если `true`, то поле скрыто.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   for filterIdx = 0, filter:getCount() - 1 do
      print(filter:isHidden(filterIdx))
   end
end
```

3.9.12.5 Метод PivotTableFilter:setHidden

Устанавливает видимость поля для заданного индекса. Параметры: `itemName` – индекс поля, `hidden` – видимость (`true` – поле скрыто).

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   for filterIdx = 0, filter:getCount() - 1 do
      print(filter:setName(filterIdx, false))
   end
end
```

3.9.13 Таблица DocumentAPI.PivotTableField

Таблица `DocumentAPI.PivotTableField` содержит свойства полей сводной таблицы (см. Таблицу 68). Таблица может быть получена посредством вызова `PivotTable:getFieldsList()`.

Таблица 68 – Описание полей таблицы `DocumentAPI.PivotTableField`

Поле	Описание
`PivotTableField.fieldProperties`	Свойства полей сводной таблицы `PivotTablFieldProperties`.
`PivotTableField.fieldCategories`	Категории полей сводной таблицы `PivotTablFieldCategories`.
`PivotTableField.customFormula`	Вычисляемая формула (строка).

3.9.14 Таблица DocumentAPI.PivotTableFieldProperties

`DocumentAPI.PivotTableFieldProperties` содержит свойства поля `DocumentAPI.PivotTableField` сводной таблицы (см. Таблицу 69).

Таблица 69 – Описание полей таблицы `DocumentAPI.PivotTableFieldProperties`

Поле	Описание
`PivotTableFieldProperties.fieldName`	Имя поля.
`PivotTableFieldProperties.fieldAlias`	Псевдоним поля (пользовательское имя).
`PivotTableFieldProperties.subtotalAlias`	Псевдоним подытогов конкретного поля.

3.9.15 Таблица DocumentAPI.PivotTableCategoryField

`DocumentAPI.PivotTableCategoryField` содержит свойства поля сводной таблицы, использующегося как строка / столбец (см. Таблицу 70). Таблица может быть получена посредством вызовов `PivotTable:getRowFields()`, `PivotTable:getColumnFields()`.

Таблица 70 – Описание полей таблицы `DocumentAPI.PivotTableCategoryField`

Поле	Описание
`PivotTableCategoryField.fieldProperties`	Свойства поля `PivotTableFieldProperties`.
`PivotTableCategoryField.subtotalFunctions`	Список функций `PivotTableFunction` для вычисления подытога.

3.9.16 Таблица DocumentAPI.PivotTableValueField

`DocumentAPI.PivotTableValueField` содержит свойства поля сводной таблицы, использующегося как значение столбец (см. Таблицу 71). Таблица может быть получена посредством вызова `PivotTable:getValueFields()`.

Таблица 71 – Описание полей таблицы `DocumentAPI.PivotTableValueField`

Поле	Описание
`PivotTableValueField baseFieldName`	Оригинальное поле на основе которого было создано данное поле, тип - строка.
`PivotTableValueField valueFieldName`	Автоматический уникальный псевдоним такой как “Sum of %имя поля%“, тип – строка.
`PivotTableValueField cellNumberFormat`	Числовой формат типа `CellFormat` для конкретного поля значений.
`PivotTableValueField totalFunction`	Агрегирующая функция `PivotTableFunction` поля значений (SUM, COUNT, MAX и т.д).

Поле	Описание
`PivotTableValueField customFormula`	Вычисляемая формула для поля значений, тип - строка.

3.9.17 Таблица DocumentAPI.PivotTablePageField

Содержит свойства поля из области фильтров (см. Таблицу 72). Таблица может быть получена посредством вызова `PivotTable:getPageFields()`.

Таблица 72 – Описание полей таблицы `DocumentAPI.PivotTablePageField`

Поле	Описание
`PivotTablePageField.fieldProperties`	Свойства поля `PivotTableFieldProperties`.

3.9.18 Таблица DocumentAPI.PivotTableItems

Таблица обеспечивает доступ к списку элементов сводной таблицы (см. Рисунок 1135).

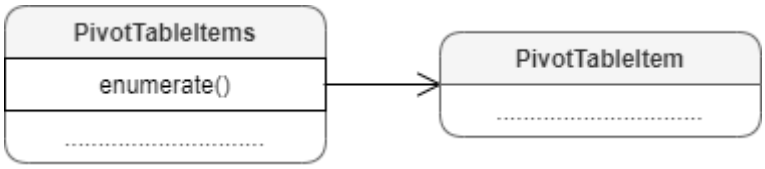


Рисунок 1135 – Объектная модель таблиц для работы с элементами сводных таблиц

3.9.18.1 Метод PivotTableItems:enumerate

Используется для перечисления элементов сводной таблицы.

Пример:

```
local fieldItems = pivotTable:getFieldItems("Age")
for fieldItem in fieldItems:enumerate() do
   print(fieldItem:getName())
   print(fieldItem:getAlias())
   print(fieldItem:getItemType())
   print(fieldItem:isCollapsed())
end
```

3.9.19 Таблица DocumentAPI.PivotTableItem

`DocumentAPI.PivotTableItem` описывает элемент сводной таблицы (см. Рисунок 1136). См. пример в главе `PivotTableItems:enumerate`.

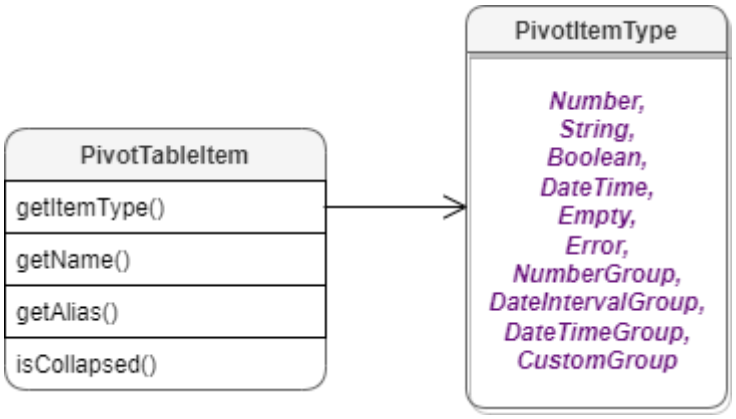


Рисунок 1136 – Таблица `DocumentAPI.PivotTableItem`

3.9.19.1 Метод PivotTableItem:getName

Метод возвращает имя элемента сводной таблицы, тип - строка. См. пример в главе `PivotTableItems:enumerate`.

3.9.19.2 Метод PivotTableItem:getAlias

Метод возвращает псевдоним элемента (идентификатор, созданный пользователем), тип - строка. См. пример в главе `PivotTableItems:enumerate`.

3.9.19.3 Метод PivotTableItem:getItemType

Метод возвращает тип `DocumentAPI.PivotTableItemType` элемента сводной таблицы. См. пример в главе `PivotTableItems:enumerate`.

3.9.19.4 Метод PivotTableItem:isCollapsed

Метод возвращает `true`, если элемент сводной таблицы свернут. См. пример в главе `PivotTableItems:enumerate`.

3.9.20 Таблица DocumentAPI.PivotTableItemType

Таблица `DocumentAPI.PivotTableItemType` содержит возможные типы элементов сводной таблицы. Описание полей таблицы представлено в Таблице 73.

Таблица 73 – Описание полей таблицы `DocumentAPI.PivotTableItemType`

Поле	Описание
`DocumentAPI.PivotTableItemType_Number`	Числовой.
`DocumentAPI.PivotTableItemType_String`	Строковый.
`DocumentAPI.PivotTableItemType_Boolean`	Логический.
`DocumentAPI.PivotTableItemType_DateTime`	Дата / время.

Поле	Описание
DocumentAPI.PivotTableItemType_Empty	Пустой тип.
DocumentAPI.PivotTableItemType_Error	Ошибка.
DocumentAPI.PivotTableItemType_NumberGroup	Интервальная группировка.
DocumentAPI.PivotTableItemType_DateIntervalGroup	Интервальная группировка по датам.
DocumentAPI.PivotTableItemType_DateTimeGroup	Группировка по дате / времени.
DocumentAPI.PivotTableItemType_CustomGroup	Пользовательская (произвольная) группировка.

Пример:

```
local fieldItems = pivotTable:getFieldItems("Age")
for fieldItem in fieldItems:enumerate() do
   if (fieldItem:getItemType() == DocumentAPI.PivotTableItemType_Number) then
      print("Numeric type")
   end
end
```

3.9.21 Таблица DocumentAPI.PivotTableEditor

Предназначена для редактирования сводных таблиц. Возвращается посредством метода PivotTable:createPivotTableEditor().

3.9.21.1 Метод PivotTableEditor:addField

Метод добавляет новое поле в сводную таблицу, используя параметры:

– fieldName - имя поля;

– toCategory - категория поля (тип - DocumentAPI.PivotTableFieldCategory);

– index - позиция в категории.

Метод возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
pivotTableEditor = pivotTableEditor:addField("CC", DocumentAPI.PivotTableFieldCategory_Values)
pivotTableEditor:apply()
```

3.9.21.2 Метод PivotTableEditor:moveField

Метод перемещает поле между категориями.

Параметры:

– `fieldName` - имя поля;

– `toCategory` - область, в которую перемещается поле (тип - `DocumentAPI.PivotTableFieldCategory`);

– `index` - позиция в новой категориии.

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor:moveField("BB", DocumentAPI.PivotTableFie
ldCategory_Values, 0)
pivotTableEditor:apply()
```

3.9.21.3 Метод PivotTableEditor:removeField

Метод удаляет поле из категории.

Параметры:

– `fieldName` - имя поля,

– `fromCategory` - область, из которой удаляется поле (тип - `DocumentAPI.PivotTableFieldCategory`).

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor:removeField("Age", DocumentAPI.PivotTable
FieldCategory_Values)
pivotTableEditor:apply()
```

3.9.21.4 Метод PivotTableEditor:reorderField

Метод изменяет позицию поля в пределах категории.

Параметры:

– `fieldName` - имя поля;

– `category` - область (тип - `DocumentAPI.PivotTableFieldCategory`);

– `toIndex` - новая позиция поля.

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor:reorderField("Age", DocumentAPI.PivotTabl
eFieldCategory_Values, 0)
pivotTableEditor:apply()
```

3.9.21.5 Метод PivotTableEditor:enableField

Метод добавляет поле в область, зависящую от типа поля. Параметр `fieldName` - имя поля. Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
local pivotTableEditor = pivotTable:createPivotTableEditor()
pivotTableEditor:enableField("Age")
pivotTableEditor:apply()
```

3.9.21.6 Метод PivotTableEditor:disableField

Метод удаляет поле из всех областей. Параметр `fieldName` - имя поля (тип - строка). Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
local pivotTableEditor = pivotTable:createPivotTableEditor()
pivotTableEditor:disableField("Age")
pivotTableEditor:apply()
```

3.9.21.7 Метод PivotTableEditor:setSummarizeFunction

Метод задает суммирующую функцию для поля из области значений.

Параметры:

– `valueFieldName` - имя поля (тип - строка);

– `summarizeFunction` – суммирующая функция, тип - `DocumentAPI.PivotTableFunction`.

Метод возвращает объект `DocumentAPI.PivotTableEditor`.

Пример:

```
pivotTableEditor = pivotTableEditor:addField("Age", DocumentAPI.PivotTableFie
ldCategory_Values)
pivotTableEditor:apply()
```

3.9.21.8 Метод PivotTableEditor:setFilter

Метод задает фильтр DocumentAPI.PivotTableFilter сводной таблицы. Если фильтр не может быть применен, вызывается исключение PivotTableError. Метод возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   for filterIdx = 0, filter:getCount() - 1 do
      filter:setHidden(filterIdx, false)
      pivotTableEditor:setFilter(filter)
   end
end
pivotTableEditor:apply()
```

3.9.21.9 Метод PivotTableEditor:setFilters

Метод задает фильтры DocumentAPI.PivotTableFilters сводной таблицы. Если какой-то из фильтров не может быть применен, он пропускается. Метод возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
local filters = pivotTable:getFilters()
for filter in filters:enumerate() do
   for filterIdx = 0, filter:getCount() - 1 do
      filter:setHidden(filterIdx, false)
   end
end
pivotTableEditor:setFilter(filters)
pivotTableEditor:apply()
```

3.9.21.10 Метод PivotTableEditor:setCaptions

Метод задает заголовки сводной таблицы DocumentAPI.PivotTableCaptions, возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
local pivotTableCaptions = pivotTable:getPivotTableCaptions()
pivotTableCaptions.grandTotalCaption = "Общий итог за год"


local pivotTableEditor = pivotTable:createPivotTableEditor()
```

```
pivotTableEditor = pivotTableEditor:setCaptions(pivotTableCaptions)
pivotTableEditor:apply()
```

3.9.21.11 Метод PivotTableEditor:setLayoutSettings

Метод устанавливает настройки отображения DocumentAPI.PivotTableLayoutSettings сводной таблицы, возвращает объект DocumentAPI.PivotTableEditor.

Пример:

```
local layoutSettings = pivotTable:getPivotTableLayoutSettings()
layoutSettings.reportLayout = DocumentAPI.PivotTableReportLayout_Tabul
ar


local pivotTableEditor = pivotTable:createPivotTableEditor()
pivotTableEditor = pivotTableEditor:setLayoutSettings(layoutSettings)
pivotTableEditor:apply()
```

3.9.21.12 Метод PivotTableEditor:setGrandTotalSettings

Метод задает настройки отображения общего итога.

Параметры:

– `isRowGrandTotalEnabled` – показывать общие итоги для строк;

– `isColGrandTotalEnabled` – показывать общие итоги для столбцов.

Пример:

```
local pivotTableEditor = pivotTable:createPivotTableEditor()
pivotTableEditor:setGrandTotalSettings(true, true)
```

3.9.21.13 Метод PivotTableEditor:apply

Метод обновляет сводную таблицу с заданными свойствами и возвращает результат DocumentAPI.PivotTableUpdateResult.

Пример:

```
local pivotTableEditor = pivotTable:createPivotTableEditor()
if DocumentAPI.PivotTableUpdateResult_Success == pivotTableEditor:apply() the
n
    print("Successfully applied");
end
```

3.9.22 Таблица DocumentAPI.PivotTableUpdateResult

В Таблице 74 приведены константы, которые соответствуют возможным результатам обновления сводной таблицы (см. методы `PivotTable.update()`, `PivotTableEditor:apply()`).

Таблица 74 – Результаты обновления сводной таблицы

Наименование константы	Описание
`DocumentAPI.PivotTableUpdateResult_Success`	Успешное обновление таблицы
`DocumentAPI.PivotTableUpdateResult_NoPivotTable`	Сводная таблица не найдена
`DocumentAPI.PivotTableUpdateResult_NoSuchFieldInCategory`	Не найдено поле в категории
`DocumentAPI.PivotTableUpdateResult_NoSuchFieldInPivotTable`	Не найдено поле в сводной таблице
`DocumentAPI.PivotTableUpdateResult_InvalidIndex`	Ошибка в индексе
`DocumentAPI.PivotTableUpdateResult_FieldAlreadyEnabled`	Поле уже существует
`DocumentAPI.PivotTableUpdateResult_MovingFieldToTheSameCategoryForbidden`	Попытка перемещения поля в рамках текущей категории
`DocumentAPI.PivotTableUpdateResult_InvalidFunction`	Неправильная функция
`DocumentAPI.PivotTableUpdateResult_InvalidCategory`	Неправильная область
`DocumentAPI.PivotTableUpdateResult_InvalidDataSourceRange`	Ошибка диапазона исходных данных
`DocumentAPI.PivotTableUpdateResult_NoDataRowsInDataSource`	В исходных данных нет строк с данными
`DocumentAPI.PivotTableUpdateResult_EmptyDataSourceHeaders`	Пустые заголовки исходных данных
`DocumentAPI.PivotTableUpdateResult_NoReferenceUnderDefine`	Попытка обновить или создать сводную таблицу на именованном диапазоне который не содержит ссылку, а содержит константу
`DocumentAPI.PivotTableUpdateResult_NoSuchItem`	Элемент не найден
`DocumentAPI.PivotTableUpdateResult_CannotExpandCollapseLeafItem`	Не удается раскрыть свернутый элемент

Наименование константы	Описание
DocumentAPI.PivotTableUpdateResult_AnotherPivotInsideDataSource	Найдена другая сводная таблица в этом же диапазоне
DocumentAPI.PivotTableUpdateResult_Canceled	Обновление сводной таблицы отменено

3.9.23 Таблица DocumentAPI.PivotTableFieldCategory

Таблица DocumentAPI.PivotTableFieldCategory описывает флаги, которые задают категорию области полей. Описание полей таблицы представлено в Таблице 75.

Таблица 75 – Описание полей таблицы DocumentAPI.PivotTableFieldCategory

Поле	Описание
DocumentAPI.PivotTableFieldCategory_Pages	Область фильтров
DocumentAPI.PivotTableFieldCategory_Rows	Область строк
DocumentAPI.PivotTableFieldCategory_Columns	Область колонок
DocumentAPI.PivotTableFieldCategory_Values	Область значений

3.10 Графические объекты

Редакторы текста и таблиц МойОфис поддерживают несколько типов графических объектов со схожим поведением: изображения (DocumentAPI.Image) и фигуры (DocumentAPI.Shape), которые являются разновидностью фигур.

Объектная модель документа в части управления изображениями развивается и дополняется возможностями. Доступны следующие операции:

- Вставка изображений в текстовый и табличный документ. Место вставки обозначается заранее с помощью закладки или таблицы с невидимыми границами.
- Перечисление графических объектов, находящихся в документе, определение их типа и геометрических размеров.
- Перемещение графических объектов, изменение их размеров и масштаба.

Перечисление графических объектов в текстовом документе.

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
   frame = mediaObject:getFrame()
   framePosition = frame:getDimensions()
   print("Position:", framePosition)
```

```
   if mediaObject:toImage() then
     print("Изображение")
   else
     print("Фигура")
   end
end
```

Перечисление графических объектов в табличном документе

```
local tbl = document:getBlocks():getTable(0)
local mediaObjects = tbl:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
   local image = mediaObject:toImage()
   if image ~= nil then
      frame = mediaObject:getFrame()
      topLeftPos = frame:getTopLeft()
      print(topLeftPos.x, topLeftPos.y)
   end
end
```

Вставка изображения в текстовый документ

```
local range = document:getRange()
local imageSize = DocumentAPI.SizeU(50, 50)
range:getBegin():insertImage("C://Tmp/123.jpg", imageSize)
```

Вставка изображения в табличный документ

В текущей версии не поддерживается.

3.10.1 Таблица DocumentAPI.MediaObjects

Таблица `DocumentAPI.MediaObjects` предназначен для доступа к коллекции графических объектов. Может быть получена вызовом методов `Table.getMediaObjects()` или `Range.getInlineObjects()` (см. Рисунок 1137).

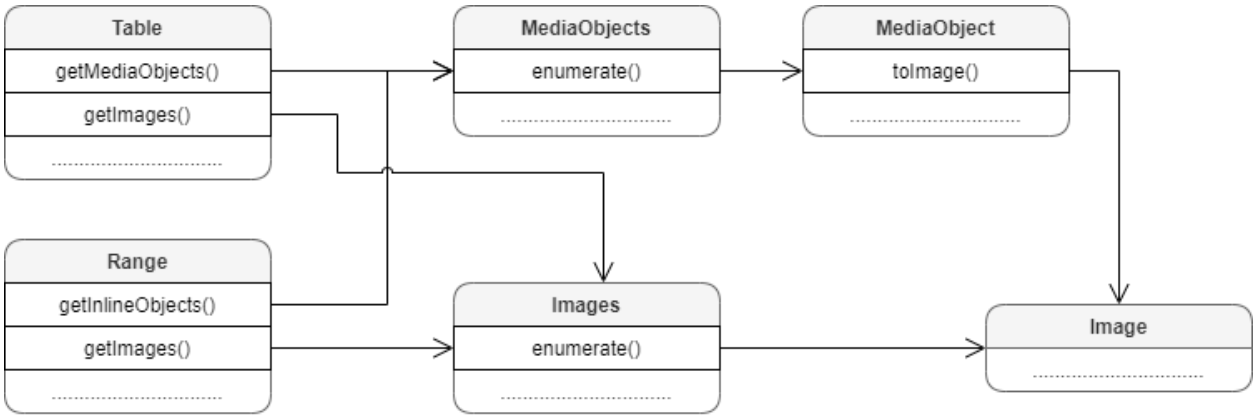


Рисунок 1137 – Графические объекты

3.10.1.1 Метод MediaObjects:enumerate

Метод позволяет перечислить коллекцию встроенных объектов.

Примеры для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
  print(mediaObject:getFrame():getWrapType())
end

for mediaObject in EditorAPI.getSelection():getInlineObjects():enumerate() do
  print(mediaObject:getFrame():getWrapType())
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
  local image = mediaObject:toImage()
  if image ~= nil then
    print("Текущий объект является изображением")
  else
    print("Текущий объект является фигурой")
  end
end
```

3.10.2 Таблица DocumentAPI.MediaObject

Таблица `DocumentAPI.MediaObject` представляет собой встроенный объект документа.

3.10.2.1 Метод MediaObject:toImage

Метод возвращает изображение DocumentAPI.Image, связанное со встроенным объектом. Если объект не является изображением, метод возвращает `nil`.

Пример для текстового документа:

```
for mediaObject in document:getRange():getInlineObjects():enumerate() do
  local image = mediaObject:toImage()
  if image then
    print("Текущий объект является изображением")
  else
    print("Текущий объект является фигурой")
  end
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
  local image = mediaObject:toImage()
  if image ~= nil then
    print("Текущий объект является изображением")
  else
    print("Текущий объект является фигурой")
  end
end
```

3.10.2.2 Метод MediaObject:getFrame

Метод возвращает свойства позиции встроенного объекта. В зависимости от текущего редактора метод возвращает разные типы таблиц. Графические объекты текстового редактора привязаны к позиции в документе, поэтому для описания местоположения и размеров используют таблицу DocumentAPI.InlineFrame, табличные документы работают с абсолютной позицией и используют таблицу DocumentAPI.AbsoluteFrame.

Пример для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
  print(mediaObject:getFrame()) -- <userdata of type 'CO::API::Document::
InlineFrame'>
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
  print(mediaObject:getFrame()) -- <userdata of type 'CO::API::Document::
AbsoluteFrame'>
end
```

3.10.3 Таблица DocumentAPI.Image

Таблица `DocumentAPI.Image` представляет собой изображение, находящееся в текстовом или табличном документе.

3.10.3.1 Метод Image:getFrame

Метод аналогичен методу `MediaObject:getFrame()`, он возвращает свойства позиции изображения. В зависимости от текущего редактора метод возвращает разные типы рамок. Графические объекты текстового редактора привязаны к позиции в документе, поэтому для описания местоположения и размеров используют тип `DocumentAPI.InlineFrame`, табличные документы работают с абсолютной позицией и используют тип `DocumentAPI.AbsoluteFrame`.

Пример для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
  local image = mediaObject:toImage()
  if (image) then
    print(image:getFrame()) -- <userdata of type 'CO::API::Document::Inli
neFrame'>
  end
end
```

Пример для табличного документа:

```
local table = document:getBlocks():getTable(0)
local mediaObjects = table:getMediaObjects()
for mediaObject in mediaObjects:enumerate() do
  local image = mediaObject:toImage()
  if (image) then
    print(image:getFrame()) -- <userdata of type 'CO::API::Document::Abso
luteFrame'>
  end
end
```

3.10.3.2 Метод Image:remove

Метод удаляет изображение из документа.

Пример для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
  local image = mediaObject:toImage()
  if (image) then
    image:remove()
    break
  end
end
```

3.10.4 Таблица DocumentAPI.Images

Таблица `DocumentAPI.Images` используется для доступа к коллекции изображений. Может быть получена вызовом методов `Table.getImages()`, `Range.getImages()`.

3.10.4.1 Метод Images:enumerate

Метод позволяет перечислить коллекцию изображений.

Пример для текстового документа:

```
for image in EditorAPI.getSelection():getImages():enumerate() do
  print(image:getFrame():getWrapType())
end
```

Пример для табличного документа:

```
local sheet = document:getBlocks():getTable(0)
local images = sheet:getImages()
for image in images:enumerate() do
  print(image:getFrame():getTopLeft().x)
end
```

3.10.5 Таблица DocumentAPI.AbsoluteFrame

Таблица `DocumentAPI.AbsoluteFrame` описывает прямоугольную область медиаобъекта, находящегося в абсолютной позиции документа (см. Рисунок 1138). Предназначена для получения и изменения свойств позиции медиаобъектов. Используется в табличном документе.

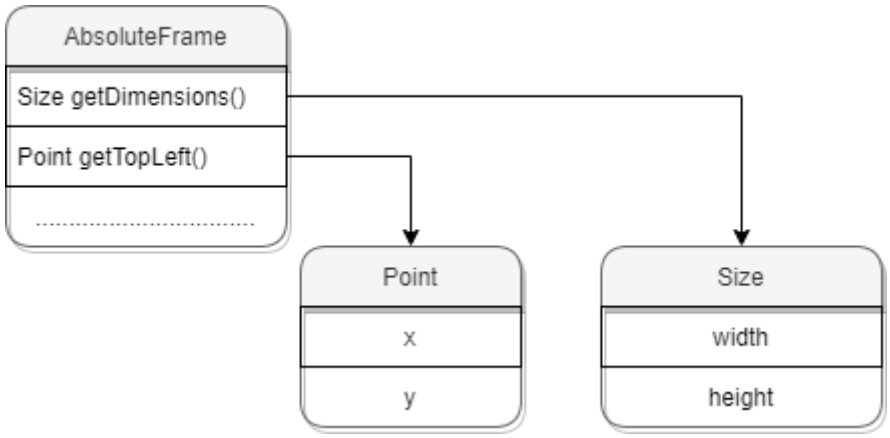


Рисунок 1138 – Объектная модель таблицы `DocumentAPI.AbsoluteFrame`

Пример для табличного документа:

```
local sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
   local absoluteFrame = mediaObject:getFrame()
   print(absoluteFrame:getDimensions())
   print(absoluteFrame:getTopLeft())
end
```

3.10.5.1 Метод AbsoluteFrame:moveTo

Метод перемещает объект в заданную позицию, тип аргумента - `DocumentAPI.PointU`.

Пример:

```
local sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
   local absoluteFrame = mediaObject:getFrame()
   newFramePosition = DocumentAPI.PointU(20, 20)
   absoluteFrame:moveTo(newFramePosition)
end
```

3.10.5.2 Метод AbsoluteFrame:getTopLeft

Метод возвращает позицию верхней левой точки медиаобъекта, тип - DocumentAPI.PointU.

Пример:

```
sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
   topLeftPosition = mediaObject:getFrame():getTopLeft()
   print("x=", topLeftPosition.x, "y=", topLeftPosition.y)
end
```

3.10.5.3 Метод AbsoluteFrame:scale

Метод `scale` изменяет размер объекта, масштабируя его по горизонтали и вертикали. Возможно изменение позиции объекта в соответствии со значением аргумента `scaleFrom`.

Вызов:

```
scale(widthScale, heightScale, scaleFrom)
```

Параметры:

- `widthScale` – коэффициент масштабирования по горизонтали, тип - числовой;
- `heightScale` – коэффициент масштабирования по вертикали, тип - числовой;
- `scaleFrom` – точка, сохраняющая позицию при масштабировании, тип - DocumentAPI.ScaleFrom.

Пример:

```
-- Уменьшение масштаба всех медиаобъектов на 50%
sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
   mediaObject:getFrame():scale(0.5, 0.5, DocumentAPI.ScaleFrom_TopLeft)
end
```

3.10.5.4 Метод AbsoluteFrame:setDimensions

Метод задает размеры (изменяет размер) медиаобъекта.

Вызов:

```
setDimensions(size)
```

Параметры:

`size` – размеры встроенного объекта, тип - DocumentAPI.SizeU.

Пример:

```
-- Изменение размера всех медиаобъектов
sheet = document:getBlocks():getTable(0)
for mediaObject in sheet:getMediaObjects():enumerate() do
   mediaObject:getFrame():setDimensions(100, 100)
end
```

3.10.5.5 Метод AbsoluteFrame:getDimensions

Возвращает размеры медиаобъекта, тип - DocumentAPI.SizeU.

3.10.6 Таблица DocumentAPI.ScaleFrom

В Таблице 76 представлены позиции объекта, остающиеся неизменными при масштабировании объекта. Используется в AbsoluteFrame.scale().

Таблица 76 – Неизменные позиции объекта при масштабировании

Наименование константы	Позиция
DocumentAPI.ScaleFrom_BottomRight	Правый нижний угол.
DocumentAPI.ScaleFrom_BottomLeft	Левый нижний угол.
DocumentAPI.ScaleFrom_TopLeft	Левый верхний угол.
DocumentAPI.ScaleFrom_TopRight	Правый верхний угол.

3.10.7 Таблица DocumentAPI.InlineFrame

Таблица `DocumentAPI.InlineFrame` описывает прямоугольную область графического объекта, находящегося в текстовой позиции документа (см. Рисунок 1139). Предназначена для получения и изменения свойств позиции графических объектов. Используется в текстовом документе.

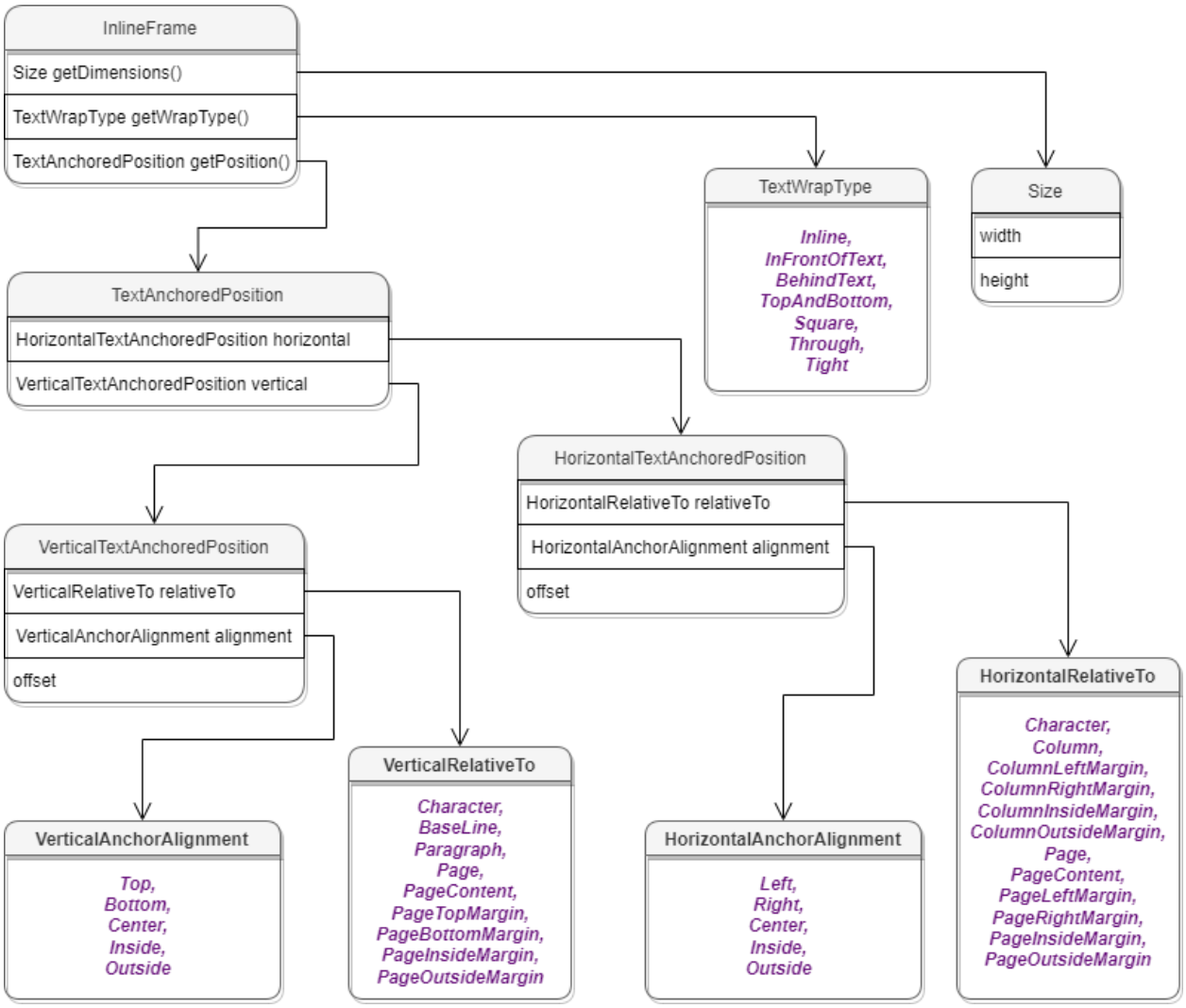


Рисунок 1139 – Объектная модель таблицы `DocumentAPI.InlineFrame`

Пример для текстового документа:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
  local inlineFrame = mediaObject:getFrame()
  print(inlineFrame:getDimensions())
  print(inlineFrame:getWrapType())
```

```
  print(inlineFrame:getPosition())
end
```

3.10.7.1 Метод InlineFrame:setPosition

Метод задает положение встроенного объекта, тип аргумента DocumentAPI.TextAnchoredPosition. Новая позиция может быть установлена только для встроенных объектов, тип переноса текста которых не является типом DocumentAPI.TextWrapType_Inline.

Пример:

```
local pos = DocumentAPI.TextAnchoredPosition()

-- Установка смещения по горизонтали относительно края колонки
pos.horizontal =
DocumentAPI.HorizontalTextAnchoredPosition(DocumentAPI.HorizontalRelativeTo_C
olumn)
pos.horizontal.offset = x

-- Установка смещения по вертикали относительно края страницы
pos.vertical =
DocumentAPI.VerticalTextAnchoredPosition(DocumentAPI.VerticalRelativeTo_Page)
pos.vertical.offset = y

-- Установка позиции рамки графического объекта
inlineFrame:setPosition(pos)
```

3.10.7.2 Метод InlineFrame:getPosition

Метод возвращает позицию встроенного объекта на странице в виде таблицы DocumentAPI.TextAnchoredPosition.

Пример:

```
local mediaObjects = document:getRange():getImages()
for mediaObject in mediaObjects:enumerate() do
  local inlineFrame = mediaObject:getFrame()
  local textAnchoredPosition = inlineFrame:getPosition()
  if (textAnchoredPosition) then
    print(textAnchoredPosition.horizontal, textAnchoredPosition.vertical)
  end
end
```

3.10.7.3 Метод InlineFrame:setDimensions

Метод задает размер DocumentAPI.SizeU встроенного объекта.

Пример:

```
inlineFrame:setDimensions(DocumentAPI.SizeU(100, 100))
```

3.10.7.4 Метод InlineFrame:getDimensions

Метод возвращает задает размеры встроенного объекта, тип - DocumentAPI.Size.

Пример:

```
local mediaObjects = document:getRange():getImages()
for mediaObject in mediaObjects:enumerate() do
  local inlineFrame = mediaObject:getFrame()
  local dimensions = inlineFrame:getDimensions()
  if (dimensions) then
    print(dimensions.width, dimensions.height)
  end
end
```

3.10.7.5 Метод InlineFrame:setWrapType

Метод устанавливает вариант обтекания текстом встроенного объекта (см. DocumentAPI.TextWrapType).

Пример:

```
local mediaObjects = document:getRange():getImages()
for mediaObject in mediaObjects:enumerate() do
  local inlineFrame = mediaObject:getFrame()
  inlineFrame:setWrapType(DocumentAPI.TextWrapType_InFrontOfText)
end
```

3.10.7.6 Метод InlineFrame:getWrapType

Метод возвращает вариант обтекания текстом встроенного объекта (см. DocumentAPI.TextWrapType).

Пример:

```
local mediaObjects = document:getRange():getImages()
for mediaObject in mediaObjects:enumerate() do
  print(mediaObject:getFrame():getWrapType())
end
```

3.10.8 Таблица DocumentAPI.TextWrapType

В Таблице 77 представлены варианты обтекания текстом встроенного объекта. Используется в `InlineFrame.setWrapType()`.

Таблица 77 – Варианты обтекания текстом встроенного объекта

Наименование константы	Описание
`DocumentAPI.TextWrapType_Inline`	Встроенный объект располагается в тексте.
`DocumentAPI.TextWrapType_InFrontOfText`	Встроенный объект располагается перед текстом.
`DocumentAPI.TextWrapType_BehindText`	Встроенный объект располагается за текстом.
`DocumentAPI.TextWrapType_TopAndBottom`	Текст располагается сверху и снизу от встроенного объекта.
`DocumentAPI.TextWrapType_Square`	Текст располагается вокруг прямоугольной рамки встроенного объекта.
`DocumentAPI.TextWrapType_Through`	Текст обтекает встроенный объект по сторонам и внутри.

3.10.9 Таблица DocumentAPI.TextAnchoredPosition

Таблица `DocumentAPI.TextAnchoredPosition` представляет позицию объекта на странице текстового документа. Описание полей таблицы представлено в Таблице 78.

Таблица 78 – Описание полей таблицы DocumentAPI.TextAnchoredPosition

Поле	Описание
`DocumentAPI.TextAnchoredPosition.horizontal`	Позиция по горизонтали `HorizontalTexAnchoredPosition`.
`DocumentAPI.TextAnchoredPosition.vertical`	Позиция по вертикали `VerticalTextAnchoredPosition`.

3.10.9.1 Метод TextAnchoredPosition:__eq

Метод используется для определения эквивалентности значений двух позиций объектов.

3.10.9.2 Метод TextAnchoredPosition:__ne

Метод используется для определения неэквивалентности значений двух позиций объектов.

3.10.10 Таблица DocumentAPI.HorizontalTextAnchoredPosition

Таблица `DocumentAPI.HorizontalTextAnchoredPosition` предназначена для управления относительным положением объекта со смещением или выравниванием по горизонтали.

Описание полей таблицы `DocumentAPI.HorizontalTextAnchoredPosition` представлено в Таблице 79.

Таблица 79 – Описание полей таблицы DocumentAPI.HorizontalTextAnchoredPosition

Поле	Описание
`DocumentAPI.HorizontalTextAnchoredPosition.relativeTo`	Тип размещения объекта относительно закрепленной позиции по горизонтали `HorizontalRelativeTo`.
`DocumentAPI.HorizontalTextAnchoredPosition.offset`	Смещение объекта.
`DocumentAPI.HorizontalTextAnchoredPosition.aligment`	Тип выравнивания объекта относительно закрепленной позиции по горизонтали `HorizontalAnchorAlignment`.

3.10.10.1 Метод HorizontalTextAnchoredPosition:__eq

Метод используется для определения эквивалентности двух положений объекта по горизонтали.

3.10.10.2 Метод HorizontalTextAnchoredPosition:__ne

Метод используется для определения неэквивалентности двух положений объекта по горизонтали.

3.10.11 Таблица DocumentAPI.VerticalTextAnchoredPosition

Таблица `DocumentAPI.VerticalTextAnchoredPosition` предназначена для управления относительным положением объекта со смещением или выравниванием по вертикали.

Описание полей таблицы `DocumentAPI.VerticalTextAnchoredPosition` представлено в Таблице 80.

Таблица 80 – Описание полей таблицы DocumentAPI.VerticalTextAnchoredPosition

Поле	Описание
`DocumentAPI.VerticalTextAnchoredPosition.rel`	Тип размещения объекта

Поле	Описание
ativeTo	относительно закрепленной позиции по вертикали VerticalRelativeTo.
DocumentAPI.VerticalTextAnchoredPosition.offset	Смещение объекта.
DocumentAPI.VerticalTextAnchoredPosition.aligment	Тип выравнивания объекта относительно закрепленной позиции по вертикали VerticalAnchorAlignment.

3.10.11.1 Метод VerticalTextAnchoredPosition:__eq

Метод используется для определения эквивалентности двух положений объекта по вертикали.

3.10.11.2 Метод VerticalTextAnchoredPosition:__ne

Метод используется для определения неэквивалентности двух положений объекта по вертикали.

3.10.12 Таблица DocumentAPI.VerticalRelativeTo

В Таблице 81 представлены типы размещения объекта относительно закрепленной позиции по вертикали.

Таблица 81 – Типы размещения объекта относительно закрепленной позиции по вертикали

Наименование константы	Описание
DocumentAPI.VerticalRelativeTo_Character	Символ.
DocumentAPI.VerticalRelativeTo_BaseLine	Базовая линия.
DocumentAPI.VerticalRelativeTo_Paragraph	Абзац.
DocumentAPI.VerticalRelativeTo_Page	Страница.
DocumentAPI.VerticalRelativeTo_PageContent	Содержимое страницы.
DocumentAPI.VerticalRelativeTo_PageTopMargin	Верхнее поле страницы.
DocumentAPI.VerticalRelativeTo_PageBottomMargin	Нижнее поле страницы.
DocumentAPI.VerticalRelativeTo_PageInsideMargin	Внутреннее поле страницы.
DocumentAPI.VerticalRelativeTo_PageOutsideMargin	Внешнее поле страницы.

3.10.13 Таблица DocumentAPI.HorizontalRelativeTo

В Таблице 82 представлены типы размещения объекта относительно закрепленной позиции по горизонтали.

Таблица 82 – Типы размещения объекта относительно закрепленной позиции по горизонтали

Наименование константы	Описание
DocumentAPI.HorizontalRelativeTo_Character	Символ.
DocumentAPI.HorizontalRelativeTo_Column	Столбец.
DocumentAPI.HorizontalRelativeTo_ColumnLeftMargin	Левое поле столбца.
DocumentAPI.HorizontalRelativeTo_ColumnRightMargin	Правое поле столбца.
DocumentAPI.HorizontalRelativeTo_ColumnInsideMargin	Внутреннее поле столбца.
DocumentAPI.HorizontalRelativeTo_ColumnOutsideMargin	Внешнее поле столбца.
DocumentAPI.HorizontalRelativeTo_Page	Страница.
DocumentAPI.HorizontalRelativeTo_PageContent	Содержимое страницы.
DocumentAPI.HorizontalRelativeTo_PageLeftMargin	Левое поле страницы.
DocumentAPI.HorizontalRelativeTo_PageRightMargin	Правое поле страницы.
DocumentAPI.HorizontalRelativeTo_PageInsideMargin	Внутреннее поле страницы.
DocumentAPI.HorizontalRelativeTo_PageOutsideMargin	Внешнее поле страницы.

3.10.14 Таблица DocumentAPI.VerticalAnchorAlignment

В Таблице 83 представлены типы выравнивания объекта относительно закрепленной позиции по вертикали.

Таблица 83 – Типы выравнивания объекта относительно закрепленной позиции по вертикали

Наименование константы	Описание
DocumentAPI.VerticalAnchorAlignment_Top	По верхнему краю.
DocumentAPI.VerticalAnchorAlignment_Bottom	По нижнему краю.
DocumentAPI.VerticalAnchorAlignment_Center	По центру.
DocumentAPI.VerticalAnchorAlignment_Inside, DocumentAPI.VerticalAnchorAlignment_Outside	По границам.

3.10.15 Таблица DocumentAPI.HorizontalAnchorAlignment

В Таблице 84 представлены типы выравнивания объекта относительно закрепленной позиции по горизонтали.

Таблица 84 – Типы выравнивания объекта относительно закрепленной позиции по горизонтали

Наименование константы	Описание
`DocumentAPI.HorizontalAnchorAlignment_Left`	По верхнему краю.
`DocumentAPI.HorizontalAnchorAlignment_Right`	По нижнему краю.
`DocumentAPI.HorizontalAnchorAlignment_Center`	По центру.
`DocumentAPI.HorizontalAnchorAlignment_Inside`, `DocumentAPI.HorizontalAnchorAlignment_Outside`	По границам.

3.10.16 Таблица DocumentAPI.Shape

Таблица `Shape` представляет собой фигуру, содержит методы для установки и получения свойств `DocumentAPI.ShapeProperties`.

3.10.16.1 Метод Shape:getShapeProperties

Метод возвращает свойства фигуры `DocumentAPI.ShapeProperties`.

Пример:

```
local shape = document:getBlocks():getShape(0)
local shape_properties = shape:getShapeProperties()
```

3.10.16.2 Метод Shape:setShapeProperties

Метод устанавливает свойства фигуры `DocumentAPI.ShapeProperties`.

Пример:

```
local shape = document:getBlocks():getShape(0)
local shape_properties = shape:getShapeProperties()
shape_properties.verticalAlignment = DocumentAPI.VerticalAlignment_Center
shape:setShapeProperties(shape_properties)
```

3.10.17 Таблица DocumentAPI.ShapeProperties

Таблица описывает свойства фигуры и содержит следующие поля:

– verticalAlignment - вертикальное выравнивание, тип DocumentAPI.VerticalAlignment;

– borderProperties - свойства границ фигуры, тип DocumentAPI.LineProperties;

– fill - свойства заполнения фигуры, тип DocumentAPI.Fill;

– shapeTextLayout - свойства текста внутри фигуры, тип DocumentAPI.ShapeTextLayout.

3.10.17.1 Поле ShapeProperties:borderProperties

Поле предназначено для установки свойств границ фигуры DocumentAPI.LineProperties.

3.10.17.2 Поле ShapeProperties:verticalAlignment

Поле предназначено для установки типа вертикального выравнивания DocumentAPI.VerticalAlignment.

3.10.17.3 Поле ShapeProperties:fill

Поле предназначено для установки свойств заполнения фигуры DocumentAPI.Fill.

3.10.17.4 Поле ShapeProperties:shapeTextLayout

Поле предназначено для установки свойств текста внутри фигуры DocumentAPI.ShapeTextLayout.

3.10.18 Таблица DocumentAPI.ShapeTextLayout

Таблица DocumentAPI.ShapeTextLayout описывает свойства текста, находящегося внутри фигуры. Описание полей представлено в Таблице 85. Используется в таблице DocumentAPI.ShapeProperties.

Таблица 85 – Описание полей таблицы DocumentAPI.ShapeTextLayout

Поле	Описание
ShapeTextLayout_DoNotAutoFit	Размещение текста в фигуре по умолчанию.
ShapeTextLayout_FitShapeExtentToText	Расширение фигуры под текст.
ShapeTextLayout_FitTextToShape	Заполнение фигуры текстом.

3.10.19 Таблица DocumentAPI.Fill

Таблица описывает свойства заполнения фигуры: цвет заполнения, путь к изображению фона.

3.10.19.1 Метод Fill:getColor

Метод возвращает цвет заполнения `DocumentAPI.Color`.

3.10.19.2 Метод Fill:getUrl

Метод возвращает путь к изображению, которое используется в качестве заполнения, тип - строка.

3.10.19.3 Метод Fill:isNoFill

Метод возвращает `true`, если заполнения нет.

3.11 Поиск в документе

Для поиска в документе необходимо создать экземпляр таблицы `DocumentAPI.Search` посредством вызова `DocumentAPI.createSearch`, затем использовать метод `Search:findText` (см. Рисунок 1140).

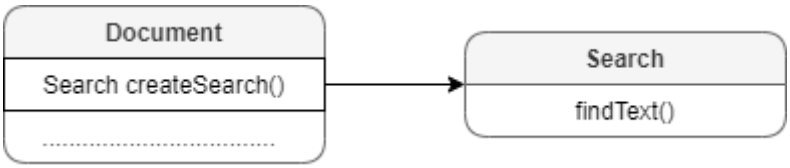


Рисунок 1140 – Объектная модель для работы с таблицами

3.11.1 Метод DocumentAPI:createSearch

Метод инициализирует механизм поиска для текущего документа. Возвращает ссылку на таблицу `DocumentAPI.Search`, с помощью методов которой выполняются поисковые запросы.

Пример:

```
search = DocumentAPI.createSearch(document)
ranges = search:findText("English")
```

3.11.2 Таблица DocumentAPI.Search

Таблица `DocumentAPI.Search` предоставляет доступ к механизму поиска и замены фрагментов документа, открытого в редакторе текста или таблиц.

3.11.2.1 Метод Search:findText

Метод выполняет поиск строки без учета регистра во всем документе или выбранном диапазоне документа. Результат возвращается в виде диапазона DocumentAPI.Range, содержащего искомый фрагмент.

Если строка не обнаружена, возвращается пустая таблица.

Примеры:

```
search = DocumentAPI.createSearch(document)
-- Поиск по всему документу
ranges = search:findText("English")
for occurrence in ranges do
    print(occurrence:extractText())
end
-- Поиск только в диапазоне первого блока
local firstBlockRange = document:getBlocks():getBlock(0):getRange()
ranges = search:findText("English", firstBlockRange)
for occurrence in ranges do
    print(occurrence:extractText())
end
```

3.12Загрузка, сохранение, экспорт, импорт документов

Метод Document.exportAs экспортирует документ в файл по указанному пути с указанным форматом.

Расширенные версии метода позволяют указать дополнительные настройки экспорта документа:

- для текстовых документов – таблица DocumentAPI.TextExportSettings;
- для табличных документов – DocumentAPI.WorkbookExportSettings.

В настоящее время поддерживается только операция экспорта документа в формат PDF/A-1b.

Пример:

```
-- Сохраняем на диск документ в формате PDF
document:exportAs(filePath, DocumentAPI.ExportFormat_PDFA1)
```

3.12.1 Таблица DocumentAPI.FormulaType

Поддерживаемые системы адресации ячеек (стили ссылок) в табличном документе представлены в Таблице 86. Используется в `document:getFormulaType()`, `document:setFormulaType()`, `DocumentAPI.DocumentSettings`.

Таблица 86 – Системы адресации ячеек в табличном документе

Наименование константы	Система адресации ячеек	Описание
`DocumentAPI.FormulaType_A1`	A1	Вариант A1 соответствует наиболее распространенной системе адресации ячеек, при которой столбцы задаются буквами, а строки – числами.
`DocumentAPI.FormulaType_R1C1`	R1C1	Вариант R1C1 соответствует альтернативной системе адресации ячеек, при которой столбцы и строки задаются числами.

3.12.2 Таблица DocumentAPI.ExportFormat

В Таблице 87 приведены поддерживаемые форматы экспорта документов (см. document:exportAs()).

Таблица 87 - Форматы экспорта документов

Наименование константы	Описание
`DocumentAPI.ExportFormat_PDFA1`	Используется для работы с документами в формате Portable Document Format (PDF) для долгосрочного архивного хранения (PDF/A-1b).

3.12.3 Таблица DocumentAPI.DocumentFormat

В Таблице 88 приведены поддерживаемые форматы документов, структура используется в `DocumentAPI.DocumentSettings`.

Таблица 88 – Форматы документов

Наименование константы	Описание
`DocumentAPI.DocumentFormat_PlainText`	Используется для работы с файлами TXT.
`DocumentAPI.DocumentFormat_DSV`	Используется для работы с табличными данными в текстовой форме (CSV, DSV). Строка текста содержит одно или несколько полей данных, разделенных запятыми или иным разделителем.
`DocumentAPI.DocumentF`	Используется для работы с текстовыми (DOCX) или

Наименование константы	Описание
ormat_OXML	табличными (XLSX) документами в формате Open Office XML.
DocumentAPI.DocumentFormat_ODF	Используется для работы с текстовыми (ODT) или табличными (ODS) документами формата Open Document Format (ГОСТ Р ИСО/МЭК 26300-2010).
DocumentAPI.DocumentFormat_HTML	Используется для работы с веб-документами в формате HTML. Работа с веб-документами в формате HTML средствами Document API в настоящий момент не поддерживается.
DocumentAPI.DocumentFormat_PDF	Используется для работы с документами в формате Portable Document Format (PDF) версии 1.4. Средствами Document API поддерживается только операция экспорта документа в формат PDF/A-1b.
DocumentAPI.DocumentFormat_PDFA	Используется для работы с документами в формате Portable Document Format (PDF) для долгосрочного архивного хранения (PDF/A-1b). Средствами Document API поддерживается только операция экспорта документа в формат PDF/A-1b.

3.12.4 Таблица DocumentAPI.DocumentSettings

Таблица DocumentAPI.DocumentSettings предоставляет общие настройки документа и используется в document:createDocument().

Описание полей таблицы DocumentAPI.DocumentSettings представлено в Таблице 89.

Таблица 89 – Описание полей таблицы DocumentAPI.DocumentSettings

Поле	Тип	Описание
DocumentAPI.DocumentSettings.documentType	DocumentType	Тип документа
DocumentAPI.DocumentSettings.userInfo	UserInfo	Информация о пользователе
DocumentAPI.DocumentSettings.localeInfo	LocaleInfo	Информация о локализации
DocumentAPI.DocumentSettings.timeZone	TimeZone	Информация о временной зоне
DocumentAPI.DocumentSettings.formulaType	FormulaType	Система адресации ячеек

3.12.5 Таблица DocumentAPI.DocumentType

В Таблице 90 приведены поддерживаемые типы документов, используется при создании документа application:createDocument(), DocumentAPI.DocumentSettings.

Таблица 90 - Типы документов

Наименование константы	Описание
DocumentAPI.DocumentType_Text	Используется для работы с текстовыми документами в форматах DOCX, ODT, XODT, TXT.
DocumentAPI.DocumentType_Workbook	Используется для работы с табличными документами в форматах XLSX, ODS, XODS.
DocumentAPI.DocumentType_Presentation	Используется для работы с презентационными документами в форматах PPTX, ODP. Работа с презентационными документами средствами Document API в настоящий момент не поддерживается.

3.12.6 Таблица DocumentAPI.SaveDocumentSettings

Таблица DocumentAPI.SaveDocumentSettings предоставляет настройки, используемые для сохранения документа в файл (см. document:saveAs()). Описание полей таблицы DocumentAPI.SaveDocumentSettings представлено в Таблице 91.

Таблица 91 – Описание полей таблицы DocumentAPI.SaveDocumentSettings

Поле	Описание
DocumentAPI.SaveDocumentSettings.documentFormat	Формат документа DocumentFormat.
DocumentAPI.SaveDocumentSettings.documentType	Тип документа DocumentType.
DocumentAPI.SaveDocumentSettings.documentPassword	Пароль для защиты электронного документа от несанкционированного доступа.
DocumentAPI.SaveDocumentSettings.isTemplate	Флаг, обозначающий, что документ должен быть сохранен как шаблон.
DocumentAPI.SaveDocumentSettings.dsvSettings	Структура DSVSettings, необходимая для сохранения в формате DSV.

3.12.7 Таблица DocumentAPI.LoadDocumentSettings

Таблица DocumentAPI.LoadDocumentSettings предоставляет дополнительные настройки, необходимые для загрузки документов из файла (см. application:loadDocument()).

Описание полей таблицы DocumentAPI.LoadDocumentSettings представлено в Таблице 73.

Таблица 92 – Описание полей таблицы DocumentAPI.LoadDocumentSettings

Поле	Описание
DocumentAPI.LoadDocumentSettings.commonDocumentSe	Экземпляр таблицы, общие

Поле	Описание
ttings	настройки документа DocumentSettings.
DocumentAPI.LoadDocumentSettings.encoding	Кодировка документа Encoding.
DocumentAPI.LoadDocumentSettings.dsvSettings	Экземпляр класса DSVSettings, настройки, необходимые для работы с файлами CSV и DSV .
DocumentAPI.LoadDocumentSettings.documentPassword	Пароль для защиты электронного документа от несанкционированного доступа. Механизм парольной защиты поддерживается только для семейства ОС Microsoft Windows.

3.12.8 Таблица DocumentAPI.Encoding

В Таблице 93 приведены поддерживаемые кодировки документов. Используется в DocumentAPI.LoadDocumentSettings.

Таблица 93 - Кодировки документов

Наименование константы	Кодировка
DocumentAPI.Encoding_Unknown	Невозможно определить кодировку.
DocumentAPI.Encoding_UTF8	UTF8
DocumentAPI.Encoding_UTF16BE	UTF16BE
DocumentAPI.Encoding_UTF16LE	UTF16LE
DocumentAPI.Encoding_UTF32BE	UTF32BE
DocumentAPI.Encoding_UTF32LE	UTF32LE
DocumentAPI.Encoding_Windows1250	Windows1250
DocumentAPI.Encoding_Windows1251	Windows1251
DocumentAPI.Encoding_Windows1252	Windows1252
DocumentAPI.Encoding_ISO8859Part5	ISO8859Part5
DocumentAPI.Encoding_KOI8R	KOI8R
DocumentAPI.Encoding_KOI8U	KOI8U
DocumentAPI.Encoding_CP866	CP866

3.12.9 Таблица DocumentAPI.TextExportSettings

Таблица `DocumentAPI.TextExportSettings` предоставляет настройки, необходимые для экспорта текстовых документов (см. document:exportAs()). Поле таблицы `DocumentAPI.TextExportSettings.pageNumbers` содержит таблицу `DocumentAPI.PageNumbers`, в которой содержатся настройки страниц для экспорта текстовых документов.

Пример:

```
textExportSettings = DocumentAPI.TextExportSettings()
textExportSettings.pageNumbers =
DocumentAPI.PageNumbers(DocumentAPI.PageParity_Even)
document:exportAs(filePath, DocumentAPI.ExportFormat_PDFA1,
textExportSettings)
```

3.12.10 Таблица DocumentAPI.WorkbookExportSettings

Таблица `DocumentAPI.WorkbookExportSettings` предоставляет настройки, необходимые для экспорта табличных документов (см. document:exportAs()).

Описание полей таблицы `DocumentAPI.WorkbookExportSettings` представлено в Таблице 94.

Таблица 94 – Описание полей таблицы `DocumentAPI.WorkbookExportSettings`

Поле	Описание
`DocumentAPI.WorkbookExportSettings.sheetNames`	Представляет коллекцию имен листов для экспорта, тип `VectorString`. Если коллекция пуста, экспортируются все листы.
`DocumentAPI.WorkbookExportSettings.printingScope`	Представляет область печати (весь документ, область печати, пользовательский диапазон и т. д.) `PrintingScope`.
`DocumentAPI.WorkbookExportSettings.pageProperties`	Представляют свойства страницы для выходного документа (высота и ширина страницы в пунктах pt) `PageProperties`.
`DocumentAPI.WorkbookExportSettings.scale`	Представляет масштаб экспорта выходного документа в процентах (например, 50,0%, 150,63%, 400,0% и т. д.).

Пример:

```
workbookSettings = DocumentAPI.WorkbookExportSettings()
workbookSettings.sheetNames = DocumentAPI.VectorString()
workbookSettings.sheetNames:push_back("Лист2")
workbookSettings.printingScope =
DocumentAPI.PrintingScope(DocumentAPI.PrintingScope.Type_PrintArea)
```

```
workbookSettings.pageProperties = DocumentAPI.PageProperties(100, 200)
workbookSettings.scale = 90
document:exportAs(filePath, DocumentAPI.ExportFormat_PDFA1, workbookSettings)
```

3.12.11 Таблица DocumentAPI.PrintingScope

Таблица `DocumentAPI.PrintingScope` содержит настройки для экспорта табличных документов. Используется в поле `printingScope` таблицы `DocumentAPI.WorkbookExportSettings`.

Позволяет создать области печати следующих типов:

- выбранная область печати либо весь документ (см. `PrintingScope.Type`);
- указанный диапазон ячеек (см. `DocumentAPI.CellRangePosition`).

Примеры:

```
-- по умолчанию - DocumentAPI.PrintingScope.Type_PrintArea
printingScope = DocumentAPI.PrintingScope()

-- область печати
printingScope = DocumentAPI.PrintingScope(DocumentAPI.PrintingScope.Type_Print
tArea)

-- диапазон ячеек
cellRangePosition = DocumentAPI.CellRangePosition(0, 0, 5, 5)
printingScope = DocumentAPI.PrintingScope(cellRangePosition)
```

3.12.11.1 Метод PrintingScope:getCellRange

Метод возвращает диапазон ячеек таблицы (см. `DocumentAPI.CellRangePosition`).

3.12.11.2 Метод PrintingScope:usePrintArea

Метод возвращает `true`, если область печати должна использоваться во время печати, экспорта и т. д.

3.12.12 Таблица DocumentAPI.PrintingScope.Type

Варианты выбора диапазона страниц для экспорта и печати представлены в Таблице 95. Используется в `DocumentAPI.PrintingScope`

Таблица 95 – Диапазон страниц для экспорта и печати

Наименование константы	Описание
DocumentAPI.PrintingScope.Type_PrintArea	Выбранная область печати (по умолчанию).
DocumentAPI.PrintingScope.Type_WholeSheet	Печать всего документа.

3.12.13 Таблица DocumentAPI.UserInfo

Таблица DocumentAPI.UserInfo предоставляет информацию о пользователе.

Описание полей таблицы DocumentAPI.UserInfo представлено в Таблице 96.

Таблица 96 – Описание полей таблицы DocumentAPI.UserInfo

Поле	Описание	Тип
DocumentAPI.UserInfo.name	Имя пользователя.	Строка
DocumentAPI.UserInfo.email	Адрес электронной почты пользователя.	Строка

3.12.14 Таблица DocumentAPI.PageNumbers

Таблица DocumentAPI.PageNumbers используется в качестве поля pageNumbers таблицы DocumentAPI.TextExportSettings и представляет собой коллекцию страниц для экспорта.

Позволяет установить следующие типы страниц для экспорта:

- нечетные, четные страницы, тип DocumentAPI.PageParity;
- список конкретных номеров страниц, тип DocumentAPI.VectorUInt;
- диапазон страниц с указанием начальной и конечной страницы.

Примеры:

```
-- четные страницы
pageNumbers = DocumentAPI.PageNumbers(DocumentAPI.PageParity_Even)

-- конкретные номера страниц
pages = DocumentAPI.VectorUInt(3)
pages[0] = 1
pages[1] = 13
pages[2] = 25
pageNumbers = DocumentAPI.PageNumbers(pages)

-- диапазон страниц
pageNumbers = DocumentAPI.PageNumbers(1, 20)
```

3.12.14.1 Метод PageNumbers:contains

Метод служит для проверки вхождения заданного номера страницы в коллекцию номеров страниц `DocumentAPI.PageNumbers`.

Пример:

```
pages = DocumentAPI.VectorUInt(3)
pages[0] = 1
pages[1] = 13
pages[2] = 25
pageNumbers = DocumentAPI.PageNumbers(pages)
print(pageNumbers:contains(13)) -- true
```

3.12.14.2 Метод PageNumbers:getLast

Метод `PageNumbers:getLast` возвращает последний номер страницы.

Пример:

```
pages = DocumentAPI.VectorUInt(3)
pages[0] = 1
pages[1] = 13
pages[2] = 25
pageNumbers = DocumentAPI.PageNumbers(pages)
print(pageNumbers:getLast()) -- 25
```

3.12.15 Таблица DocumentAPI.PageParity

Варианты выбора страниц для экспорта и печати представлены в Таблице 97. Используется в `DocumentAPI.PageNumbers`, `DocumentAPI.PrintSettings`.

Таблица 97 – Варианты выбора страниц для экспорта и печати

Наименование константы	Описание
`DocumentAPI.PageParity_Odd`	Только нечетные страницы.
`DocumentAPI.PageParity_Even`	Только четные страницы.
`DocumentAPI.PageParity_All`	Все страницы.

3.12.16 Таблица DocumentAPI.TimeZone

Таблица `DocumentAPI.TimeZone` предоставляет настройки, необходимые для экспорта текстовых документов.

Поле таблицы `DocumentAPI.TimeZone.offsetInSecondsToUTC` (числовой тип) содержит значение, с помощью которого задается смещение или разность между

временем в данном часовом поясе и временем в формате UTC (Всемирное координированное время).

3.12.17 Таблица DocumentAPI.DSVSettings

Таблица `DocumentAPI.DSVSettings` предоставляет настройки, необходимые для работы с файлами CSV (comma-separated value) и DSV (delimiter-separated value). Используется в `DocumentAPI.SaveDocumentSettings`, `DocumentAPI.LoadDocumentSettings`.

Описание полей таблицы `DocumentAPI.DSVSettings` представлено в Таблице 98.

Таблица 98 – Описание полей таблицы `DocumentAPI.DSVSettings`

Поле	Описание
`DocumentAPI.DSVSettings.autofit`	Признак необходимости автоматического подстраивания ширины столбца под размер данных в ячейке.
`DocumentAPI.DSVSettings.startBlockIndex`	Индекс блока документа сохранения.
`DocumentAPI.DSVSettings.lastBlockIndex`	Индекс блока документа для окончания сохранения.

Пример:

```
saveDocumentSettings = DocumentAPI.SaveDocumentSettings()
saveDocumentSettings.dsvSettings = DocumentAPI.DSVSettings()
saveDocumentSettings.dsvSettings.autofit = true
saveDocumentSettings.dsvSettings.startBlockIndex = 0
saveDocumentSettings.dsvSettings.lastBlockIndex = 10
```

3.12.18 Таблица DocumentAPI.LocaleInfo

Таблица `DocumentAPI.LocaleInfo` предоставляет информацию о локализации. Используется в поле `localeInfo` таблицы `DocumentAPI.DocumentSettings`.

Описание полей таблицы `DocumentAPI.LocaleInfo` представлено в Таблице 99.

Таблица 99 – Описание полей таблицы `DocumentAPI.LocaleInfo`

Поле	Описание
`DocumentAPI.LocaleInfo.localName`	Название локализации, представлено в формате **<language> <REGION>**, где языковой код соответствует стандарту ISO-639, а код региона стандарту ISO-3166.
`DocumentAPI.LocaleInfo.decimalSeparator`	Десятичный разделитель, отделяет целые и дробные части чисел.
`DocumentAPI.LocaleInfo.thousandSeparator`	Символ, разделяющий группы цифр в

Поле	Описание
	числовых значениях.
DocumentAPI.LocaleInfo.listSeparator	Символ, отделяющий элементы в списке.
DocumentAPI.LocaleInfo.currencySymbol	Символ валюты, используемой в текущей стране или регионе.
DocumentAPI.LocaleInfo.currencyFormat	Расположение знака валюты в текущем регионе, тип: DocumentAPI.CurrencySignPlacement.
DocumentAPI.LocaleInfo.shortDatePattern	Заданный «короткий» формат отображения даты в текущем регионе (например, 'm/d/yy' для en_US).
DocumentAPI.LocaleInfo.longDatePattern	Заданный «длинный» формат отображения даты в текущем регионе (например, 'dddd, mmmm d, yyyy' for en_US).
DocumentAPI.LocaleInfo.timePattern	Заданный формат отображения времени в текущем регионе (например, 'h:mm AM/PM' для en_US).

3.13 Таблица DocumentAPI.Application

Таблица DocumentAPI.Application управляет параметрами и объектами приложения. Предоставляет интерфейс для создания и загрузки документов. Допустимо использование только одного объекта Application для всего сеанса обработки документа.

3.13.1 Метод application:createDocument

Метод application.createDocument создает новый документ с типом DocumentAPI.DocumentType, либо DocumentAPI.DocumentSettings.

Примеры:

```
local settings = DocumentAPI.DocumentSettings()
local doc = application:createDocument(DocumentAPI.DocumentType_Text)
doc:saveAs("NewTextDocument.xodt")

local settings = DocumentAPI.DocumentSettings()
settings.documentType = DocumentAPI.DocumentType_Workbook
local doc = application:createDocument(settings)
doc:saveAs("NewSheetDocument.xlsx")
```

3.13.2 Метод application:loadDocument

Метод `application.loadDocument` загружает существующий текстовый или табличный документ из файла, находящего по указанному пути. Формат и тип документа определяются из расширения файла, если не указаны явно с помощью параметра.

Если аргумент содержит таблицу `DocumentAPI.LoadDocumentSettings`, то в этом случае откроется документ с заданными параметрами.

Пример:

```
local docSettings = DocumentAPI.DocumentSettings()
docSettings.documentType = DocumentAPI.DocumentType_Workbook
local loadSettings = DocumentAPI.LoadDocumentSettings()
loadSettings.commonDocumentSettings = docSettings
local doc = application:loadDocument("spreadsheet.xlsx", loadSettings)
```

3.13.3 Метод application:getMessenger

Метод `application.getMessenger` возвращает объект `DocumentAPI.Messenger`, реализующий логирование событий.

Пример:

```
local messenger = application:getMessenger()
```

3.14 Таблица DocumentAPI.document

Таблица `DocumentAPI.Document` осуществляет доступ к содержимому открытого текстового или табличного документа.

Например, данный пример предоставляет доступ к первому абзацу текстового документа:

```
local para = document:getBlocks():getParagraph(0)
```

3.14.1 Метод document:saveAs

Метод `Document.saveAs` сохраняет документ в файл по указанному пути. Формат и тип документа определяются расширением файла, если они не указаны в явном виде.

При необходимости, в качестве второго аргумента можно использовать таблицу `DocumentAPI.SaveDocumentSettings`, которая содержит формат документа `DocumentAPI.DocumentFormat`, тип документа `DocumentAPI.DocumentType` и пароль для защиты документа от несанкционированного доступа.

Примеры:

```
document:saveAs(filePath)

saveDocumentSettings = DocumentAPI.SaveDocumentSettings()


saveDocumentSettings.documentFormat = DocumentAPI.DocumentFormat_OXML
saveDocumentSettings.documentType = DocumentAPI.DocumentType_Workbook
saveDocumentSettings.documentPassword = "password"
saveDocumentSettings.isTemplate = false


saveDocumentSettings.dsvSettings = DocumentAPI.DSVSettings()
saveDocumentSettings.dsvSettings.autofit = true
saveDocumentSettings.dsvSettings.startBlockIndex = 0
saveDocumentSettings.dsvSettings.lastBlockIndex = 10


document:saveAs(filePath, saveDocumentSettings)
```

3.14.2 Метод document:exportAs

Метод `Document.exportAs` экспортирует документ в файл по указанному пути с указанным форматом.

Расширенные версии метода позволяют указать дополнительные настройки экспорта документа:

- для текстовых документов – таблица `DocumentAPI.TextExportSettings`;
- для табличных документов – таблица `DocumentAPI.WorkbookExportSettings`.

В настоящее время поддерживается только операция экспорта документа в формат PDF/A-1b.

Примеры:

```
document:exportAs(filePath, DocumentAPI.ExportFormat_PDFA1)

textExportSettings = DocumentAPI.TextExportSettings()
textExportSettings.pageNumbers =
DocumentAPI.PageNumbers(DocumentAPI.PageParity_Even)
document:exportAs(filePath, DocumentAPI.ExportFormat_PDFA1,
textExportSettings)

workbookSettings = DocumentAPI.WorkbookExportSettings()
workbookSettings.sheetNames = DocumentAPI.VectorString()
```

```
workbookSettings.sheetNames:push_back("Лист2")
workbookSettings.printingScope =
DocumentAPI.PrintingScope(DocumentAPI.PrintingScope.Type_PrintArea)
workbookSettings.pageProperties = DocumentAPI.PageProperties(100, 200)
workbookSettings.scale = 90
document:exportAs(filePath, DocumentAPI.ExportFormat_PDFA1, workbookSettings)
```

3.14.3 Метод document:merge

Метод `Document:merge` сравнивает текущий документ с другим документом, который передается в параметре типа `DocumentAPI.Document`.

Метод возвращает объект `DocumentAPI.Document`, содержащий результат сравнения в виде отслеживаемых изменений.

Пример:

```
local documentDiff = document:merge(anotherDoc)
```

3.14.4 Метод document:getBlocks

Метод предоставляет доступ к метатаблице `DocumentAPI.Blocks` и далее к отдельным фрагментам (абзацам, таблицам и т. д.), из которых состоит документ.

Пример:

```
local blocks = document:getBlocks()
```

3.14.5 Метод document:getBookmarks

Метод предоставляет доступ к таблице закладок `DocumentAPI.Bookmarks`.

Пример:

```
local bookmarks = document:getBookmarks()
```

3.14.6 Метод document:getScripts

Метод предоставляет доступ к таблице макрокоманд `DocumentAPI.Scripts`, содержащихся в документе.

Пример:

```
local scripts = document:getScripts()
```

3.14.7 Метод document:getRange

Метод предоставляет доступ ко всему диапазону `DocumentAPI.Range` документа.

Пример:

```
local range = document:getRange()
print(range:extractText())
```

3.14.8 Метод document:isChangesTrackingEnabled

Метод возвращает текущее состояние отслеживания изменений в документе (`true` - включены).

Пример:

```
local trackingChanges = "Disabled"
if document:isChangesTrackingEnabled() then
   trackingChanges = "Enabled"
end
```

3.14.9 Метод document:setChangesTrackingEnabled

Метод управляет состоянием отслеживания изменений в документе (включены или выключены).

Пример:

```
if trackingChanges == "Disabled" then
  document:setChangesTrackingEnabled(true)
  if document:isChangesTrackingEnabled() then
     trackingChanges = "Enabled"
  end
end
```

3.14.10 Метод document:getComments

Метод обеспечивает доступ к комментариям документа, возвращает таблицу DocumentAPI.Comments.

Пример:

```
local comments = document:getComments()
for comment in comments:enumerate() do
  print(comment:getRange())
  print(comment:getText())
  print(comment:getInfo().author)
  print(comment:getInfo().timeStamp)
  print(comment:isResolved())
  print(comment:getReplies())
end
```

3.14.11 Метод document:setPageProperties

Метод устанавливает свойство DocumentAPI.PageProperties в документе.

Пример:

```
local properties = DocumentAPI.PageProperties()
properties.width = 100
properties.height = 200
document:setPageProperties(properties)
```

3.14.12 Метод document:setFormulaType

Метод устанавливает поддерживаемую адресацию ячеек DocumentAPI.FormulaType документа.

Пример:

```
document:setFormulaType(DocumentAPI.FormulaType_A1)
```

3.14.13 Метод document:getFormulaType

Метод возвращает поддерживаемую адресацию ячеек DocumentAPI.FormulaType документа.

3.14.14 Метод document:setPageOrientation

Метод устанавливает альбомную, либо книжную ориентацию страниц в документе (см. DocumentAPI.PageOrientation).

Пример:

```
document:setPageOrientation(DocumentAPI.PageOrientation_Landscape)
```

3.14.15 Метод document:enumerateSections

Возвращает таблицу объектов типа DocumentAPI.Section.

Пример:

```
local sections = document:enumerateSections()
for section in sections do
  print(section:getPageProperties().width)
end
```

3.14.16 Метод document:getSections

Возвращает таблицу объектов типа DocumentAPI.Sections.

Пример:

```
local sections = document:getSections()
for section in sections:enumerate() do
    local properties = section:getPageProperties()
    print(properties.width)
    print(properties.height)
end
```

3.14.17 Метод document:setMirroredMarginsEnabled

Метод позволяет включать и выключать зеркальные поля в документе.

Пример:

```
document:setMirroredMarginsEnabled(true)
print(document:areMirroredMarginsEnabled())
```

3.14.18 Метод document:areMirroredMarginsEnabled

Возвращает состояние режима зеркальных полей в документе (разрешены или запрещены).

Пример:

```
document:setMirroredMarginsEnabled(true)
print(document:areMirroredMarginsEnabled())
```

3.14.19 Метод document:getPivotTablesManager

Возвращает объект DocumentAPI.PivotTablesManager, который используется для создания сводных таблиц. Метод может быть использован только в табличном редакторе.

Пример:

```
local pivotTablesManager = document:getPivotTablesManager()
print(pivotTablesManager)
```

3.14.20 Метод document:getNamedExpressions

Используется для получения списка именованных выражений DocumentAPI.NamedExpressions.

3.15 Таблица DocumentAPI.SaveUnsupportedFeature

Имеются свойства документа, которые могут быть утеряны при сохранении документа. Данные свойства описаны в Таблице 100.

Таблица 100 - Форматы неподдерживаемых свойств документа

Наименование константы	Описание
SaveUnsupportedFeature_CommentsInHeaderFooter	Комментарии в колонтитулах.
SaveUnsupportedFeature_CommentsInFootnote	Комментарии в сносках.
SaveUnsupportedFeature_CommentsInFootnote	Параграфы с разным выравниванием в заметках.

3.16 Таблица DocumentAPI.ColorRGBA

Таблица DocumentAPI.ColorRGBA предназначена для настройки цвета текста и линий. Используется четырехканальный формат, содержащий данные для красного (r), голубого (b), зеленого (g) цветов и альфа-канала (a).

Описание таблицы DocumentAPI.ColorRGBA представлено в Таблице 101.

Таблица 101 – Описание таблицы DocumentAPI.ColorRGBA

Цвет	Описание
r	Значение от 0 до 255 для установки интенсивности красного цвета.
g	Значение от 0 до 255 для установки интенсивности зеленого цвета.
b	Значение от 0 до 255 для установки интенсивности голубого цвета.
a	Значение от 0 до 255 для регулировки прозрачности. Значение 255 соответствует полностью непрозрачному цвету.

Пример:

```
local line_prop = DocumentAPI.LineProperties()
line_prop.color = DocumentAPI.Color(DocumentAPI.ColorRGBA(55, 146, 179, 200))
```

3.16.1 Метод ColorRGBA:__eq

Метод используется для определения эквивалентности двух значений цвета DocumentAPI.ColorRGBA.

3.16.2 Метод ColorRGBA:__ne

Метод используется для определения неэквивалентности двух двух значений цвета DocumentAPI.ColorRGBA.

3.17 Таблица DocumentAPI.TextProperties

Таблица DocumentAPI.TextProperties предназначена для форматирования текста. Описание полей таблицы DocumentAPI.TextProperties представлено в

Таблице 102. Свойства `DocumentAPI.TextProperties` применяются к диапазону текста `DocumentAPI.Range` (методы `Range.getTextProperties()`, `Range.setTextProperties()`).

Таблица 102 – Описание полей таблицы `DocumentAPI.TextProperties`

Поле	Тип	Описание
`TextProperties.fontName`	Строковое	Наименование шрифта, использованного для оформления фрагмента документа.
`TextProperties.fontSize`	Числовое	Размер шрифта, использованного для оформления фрагмента документа.
`TextProperties.bold`	Логическое	Значение `true` устанавливает жирное начертание для указанного фрагмента текста.
`TextProperties.italic`	Логическое	Значение `true` устанавливает начертание курсивом для указанного фрагмента текста.
`TextProperties.underline`	Логическое	Значение `true` устанавливает подчеркивание для указанного фрагмента текста.
`TextProperties.strikethrough`	Логическое	Значение `true` устанавливает начертание «зачеркнутый» для указанного фрагмента текста.
`TextProperties.allCapitals`	Логическое	Значение `true` устанавливает все буквы указанного фрагмента текста как прописные. Значение `false` устанавливает все буквы указанного фрагмента текста как строчные.
`TextProperties.scriptPosition`	`ScriptPosition`	Устанавливает отображение символа в виде надстрочного, подстрочного знака или в нормальном режиме.
`TextProperties.textColor`	`Color`	Цвет указанного фрагмента документа.
`TextProperties.backgroundColor`	`ColorRGBA`	Цвет фона указанного фрагмента документа.
`TextProperties.characterSpacing`	Числовое	Размер межсимвольного интервала.

Пример:

```
local props = DocumentAPI.TextProperties()
props.fontName = "XO Oriel"
props.fontSize = 20

-- доступ к тексту третьего абзаца
local range = document:getBlocks():getParagraph(2):getRange()

-- установить свойства фрагмента текста
range:setTextProperties(props)
```

3.18 Таблица DocumentAPI.ScriptPosition

Варианты представления текста в виде надстрочных или подстрочных знаков при работе в текстовом редакторе представлены в Таблице 103. Используется в качестве поля `scriptPosition` таблицы `DocumentAPI.TextProperties`.

Таблица 103 – Типы надстрочного и подстрочного форматирования

Наименование константы	Описание
`DocumentAPI.ScriptPosition_SuperScript`	Надстрочный знак (верхний индекс).
`DocumentAPI.ScriptPosition_SubScript`	Подстрочный знак (нижний индекс).
`DocumentAPI.ScriptPosition_NormalScript`	Без указания индекса.

Пример:

```
local props = DocumentAPI.TextProperties()
props.scriptPosition = DocumentAPI.ScriptPosition_SuperScript
range:setTextProperties(props)
```

3.19 Таблица DocumentAPI.Range

Таблица `DocumentAPI.Range` предоставляет доступ к диапазону документа. На Рисунке 1141 изображена объектная модель таблиц, относящихся к работе с диапазонами.

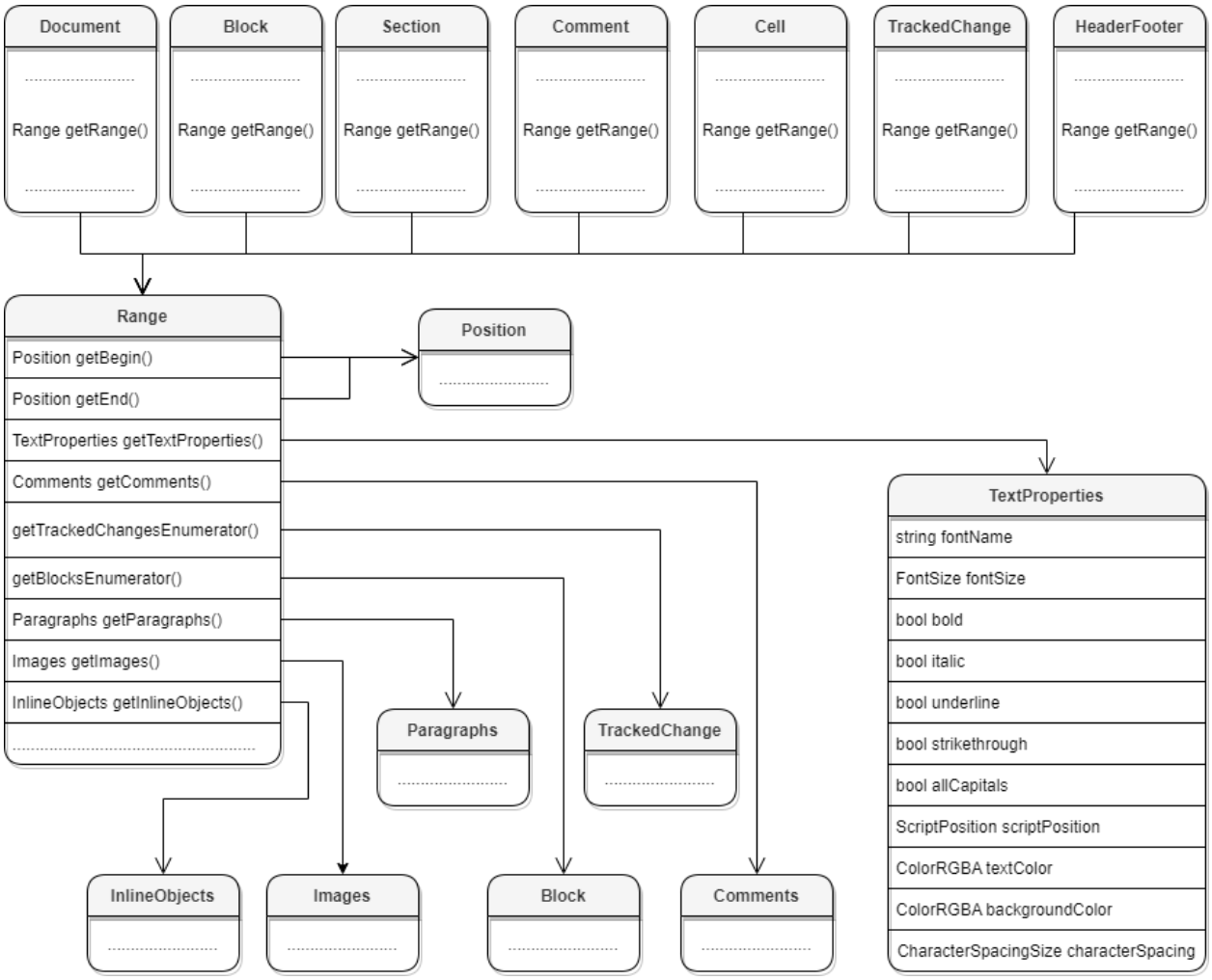


Рисунок 1141 – Объектная модель для работы с таблицей `DocumentAPI.Range`

Варианты получения диапазона для текстового документа:

```
-- диапазон всего документа
documentRange = document:getRange()
-- диапазон блока
block = document:getBlocks():getBlock(0)
blockRange = block:getRange()
-- диапазон секций
sections = document:getSections()
for section in sections:enumerate() do
    sectionRange = section:getRange()
end
-- диапазон комментариев
commentsList = document:getRange():getComments()
for comment in commentsList:enumerate() do
    commentRange = comment:getRange()
```

```
end
-- диапазон ячейки
table = document:getBlocks():getTable(0)
cell = table:getCell("B2")
cellRange = cell:getRange()
-- диапазон верхних колонтитулов
section = document:getBlocks():getBlock(0):getSection()
headers = section.getHeaders()
for header in headers:enumerate() do
    headerRange = header:getRange()
end
-- диапазон отслеживаемых изменений
local trackedChangesList = document:getRange():enumerateTrackedChanges()
for trackedChange in trackedChangesList do
    trackedChangeRange = trackedChange:getRange()
end
```

3.19.1 Метод Range:getBegin

Метод возвращает позицию в начале диапазона.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
local pos = range:getBegin() -- в начало документа
pos:insertText("Привет")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки B2
local pos = range:getBegin() -- в начало ячейки
pos:insertText("Привет")
```

3.19.2 Метод Range:getEnd

Метод возвращает позицию в конце диапазона, не включая последний символ paragraph mark.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
local pos = range:getEnd() -- в конец документа
pos:insertText("Привет")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки B2
local pos = range:getEnd() -- в конец ячейки
pos:insertText("Привет")
```

3.19.3 Метод Range:extractText

Метод возвращает содержимое фрагмента в виде строки текста. Находящиеся внутри области изображения, таблицы и другие объекты игнорируются.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
local text = range:extractText()
print (text)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки B2
print (range:extractText())
```

3.19.4 Метод Range:removeContent

Метод полностью удаляет содержимое диапазона.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:removeContent()
print (range:extractText())
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:removeContent()
print (range:extractText())
```

3.19.5 Метод Range:lockContent

Метод запрещает изменения содержимого диапазона.

Внимание ! Метод может быть использован только в текстовых документах.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:lockContent()
```

Пример для таблицы внутри текстового документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:lockContent()
```

3.19.6 Метод Range:unlockContent

Метод разрешает изменения содержимого диапазона.

Внимание ! Метод может быть использован только в текстовых документах.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:unlockContent()
```

Пример для таблицы внутри текстового документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки
range:unlockContent()
```

3.19.7 Метод Range:isContentLocked

Метод возвращает значение `true`, если изменения содержимого диапазона запрещены.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
if range:isContentLocked() then
  print("Документ содержит заблокированное содержимое")
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
```

```
local range = cell:getRange() -- содержимое ячейки
if range:isContentLocked() then
  print("Ячейка содержит заблокированное содержимое")
end
```

3.19.8 Метод Range:replaceText

Метод заменяет содержимое фрагмента на указанный текст.

Пример для текстового документа:

```
local range = document:getRange() -- содержимое всего документа
range:replaceText("Новый текст")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange() -- содержимое ячейки таблицы
range:replaceText("Новый текст")
```

3.19.9 Метод Range:setHyperlink

Метод `setHyperlink` вставляет ссылку в содержимое диапазона и заменяет его текст текстом ссылки.

Вызов:

```
setHyperlink( url, label )
```

Параметры:

– `url` – адрес ссылки;

– `label` – текст ссылки.

Пример для текстового документа:

```
local range = document:getRange()
range:setHyperlink("https://testhyperlink.com", "Hyperlink")
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
range:setHyperlink("https://testhyperlink.com", "Hyperlink")
print(cell:getFormattedValue())
```

3.19.10 Метод Range:getTextProperties

Метод возвращает таблицу с текущими настройками форматирования для фрагмента текстового документа. Описание настроек форматирования осуществляется с помощью таблицы DocumentAPI.TextProperties.

Пример для текстового документа:

```
local range = document:getRange()
local props = range:getTextProperties()
print(props.italic)
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local props = range:getTextProperties()
print(props.italic)
```

3.19.11 Метод Range:setTextProperties

Метод применяет настройки форматирования DocumentAPI.TextProperties для диапазона.

Пример для текстового документа:

```
local range = document:getRange()
local props = range:getTextProperties()
props.italic = true
range:setTextProperties(props) -- текстовый фрагмент оформлен курсивом
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local props = range:getTextProperties()
props.italic = true
range:setTextProperties(props)
```

3.19.12 Метод Range:enumerateBlocks

Предоставляет возможность итерации по блокам.

Пример для текстового документа:

```
local range = document:getRange()
for block in range:enumerateBlocks() do
```

```
    print(block:getRange():extractText())
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
for block in range:enumerateBlocks() do
    print(block:getRange():extractText())
end
```

3.19.13 Метод Range:enumerateTrackedChanges

Предоставляет возможность итерации по отслеживаемым изменениям DocumentAPI.TrackedChange. Метод может быть использован только в текстовых документах.

Пример:

```
local changesList = document:getRange():enumerateTrackedChanges()
for change in changesList do
  print(change:getRange():extractText())
end
```

3.19.14 Метод Range:getComments

Обеспечивает доступ к комментариям в диапазоне.

Комментарии, примененные к одному и тому же диапазону, упорядочиваются по датам. Если дат нет, то порядок комментариев не определен.

Пример:

```
local comments = document:getRange():getComments()
for comment in comments:enumerate() do
  print(comment:getRange())
  print(comment:getText())
  print(comment:getInfo().author)
  print(comment:getInfo().timeStamp)
  print(comment:isResolved())
  print(comment:getReplies())
end
```

3.19.15 Метод Range:getParagraphs

Обеспечивает доступ к абзацам DocumentAPI.Paragraphs в диапазоне.

Пример для текстового документа:

```
local paragraphs = document:getRange():getParagraphs()
for para in paragraphs:enumerate() do
    print(para:getRange():extractText())
end
```

Пример для табличного документа:

```
local tbl = document:getBlocks():getTable(0)
local cell = tbl:getCell("B2")
local range = cell:getRange()
local paragraphs = range:getParagraphs()

for para in paragraphs:enumerate() do
    print(para:getRange():extractText())
end
```

3.19.16 Метод Range:getImages

Обеспечивает доступ к изображениям (DocumentAPI.Image) в диапазоне.

Примеры:

```
local images = document:getRange():getImages()
for image in images:enumerate() do
  print(image:getFrame():getWrapType())
end

for image in EditorAPI.getSelection():getImages():enumerate() do
  print(image:getFrame():getWrapType())
end
```

3.19.17 Метод Range:getInlineObjects

Обеспечивает доступ к перечислению DocumentAPI.MediaObjects графических объектов диапазона.

Пример:

```
local mediaObjects = document:getRange():getInlineObjects()
for mediaObject in mediaObjects:enumerate() do
   print(mediaObject)
end
```

3.20 Таблица DocumentAPI.Position

Таблица `DocumentAPI.Position` представляет местоположение в текстовом документе. Используется для обозначения начала и конца диапазона `DocumentAPI.Range`.

3.20.1 Метод Position:insertText

Метод предназначен для вставки текстовой строки в заданное местоположение в документе.

Пример:

```
local rng = document:getRange()
local begin_pos = rng:getBegin()
begin_pos:insertText("Текст в начале строки")
```

3.20.2 Метод Position:insertTable

Метод предназначен для вставки таблицы с заданным числом строк и столбцов в заданное местоположение в документе. Возвращает объект таблицы.

Следует учитывать, что при вставке таблицы к ее имени автоматически добавляется порядковый номер, начинающийся с единицы. Таким образом, вызов

```
t = position:insertTable(3, 3, "Table")
```

приведет к созданию в текстовом документе таблицы с именем «Table1».

Пример вставки таблицы в текстовый документ:

```
local rng = document:getRange()
local begin_pos = rng:getBegin()
t = begin_pos:insertTable(3, 3, "Table")
```

В табличном документе данный метод используется для вставки нового рабочего листа.

Пример вставки нового листа в табличный документ:

```
local rng = document:getRange()
local end_pos = rng:getEnd()
t = end_pos:insertTable(3, 3, "Table")
```

3.20.3 Метод Position:insertPageBreak

Метод предназначен для вставки разрыва страницы в указанную позицию в документе.

Пример:

```
local rng = document:getRange()
local end_pos = rng:getEnd()
end_pos:insertPageBreak()
```

3.20.4 Метод Position:insertLineBreak

Метод предназначен для вставки перевода строки.

Пример:

```
local rng = document:getRange()
local end_pos = rng:getEnd()
```

3.20.5 Метод Position:insertBookmark

Вставляет закладку с наименованием в текущую позицию.

Пример:

```
document:getRange():getEnd():insertBookmark("Bookmark example")
```

3.20.6 Метод Position:insertSectionBreak

Вставляет разрыв раздела в текущую позицию.

3.20.7 Метод Position:insertHyperlink

Метод `insertHyperlink` вставляет ссылку в текущую позицию. В качестве параметров передаются адрес ссылки и текст ссылки.

Вызов:

```
insertHyperlink( url, size )
```

Параметры:

– `url` – адрес ссылки;

– `label` – текст ссылки.

Пример:

```
document:getRange():getBegin():insertHyperlink("https://testhyperlink.com",
"Hyperlink")
```

3.20.8 Метод Position:insertImage

Вставляет изображение из файла в текущую позицию.

Вызов:

```
insertImage( url, size )
```

Параметры:

– `url` – полный путь к файлу;

– `size` – геометрические размеры изображения для вставки.

Пример:

```
document:getRange():getBegin():insertImage("C://Tmp//123.jpg",
DocumentAPI.SizeU(100, 100))
```

3.20.9 Метод Position:removeBackward

Метод удаляет `count` объектов (символов, картинок и т.д.) до текущей позиции.

Пример:

```
document:getRange():getEnd():removeBackward(3)
```

3.20.10 Метод Position:removeForward

Метод удаляет `count` объектов (символов, картинок и т.д.) после текущей позиции.

Пример:

```
document:getRange():getBegin():removeForward(3)
```

3.20.11 Метод Position:__eq

Метод используется для определения эквивалентности значений двух местоположений в документе.

3.20.12 Метод Position:__ne

Метод используется для определения неэквивалентности значений двух местоположений в документе.

3.21 Таблица DocumentAPI.Messenger

3.21.1 Метод Messenger:subscribe

Метод служит для подписки на события лога.

3.21.2 Метод Messenger:notify

Метод используется для создания события лога

3.22 Таблица DocumentAPI.Connection

Таблица `Connection` реализует соединение между `DocumentAPI.Messenger` и клиентом. Содержит один метод `unsubscribe` для разрыва соединения.

3.23 Таблица DocumentAPI.Message

Таблица `Message` предназначен для формирования событий лога.

3.23.1 Таблица Message.Severity

Таблица `Message.Severity` (Таблица 104) описывает уровни сообщений лога (информация, предупреждение, ошибка).

Таблица 104 – Описание уровней лога `Message.Severity`

Поле	Описание
`DocumentAPI.Message.Severity_Info`	Информация
`DocumentAPI.Message.Severity_Warning`	Предупреждение
`DocumentAPI.Message.Severity_Error`	Ошибка

3.23.2 Метод Message:__eq

Метод используется для определения эквивалентности двух объектов `Message`.

3.23.3 Метод Message:__ne

Метод используется для определения неэквивалентности двух объектов `Message`.

3.23.4 Метод Message:getSeverity

Метод возвращает уровень лога `DocumentAPI.Message_Severity`.

3.23.5 Метод Message:getText

Метод возвращает текст сообщения.

3.23.6 Метод Message:makeInfo

Метод создает сообщение типа `DocumentAPI.Message.Severity_Info` с заданным текстом.

3.23.7 Метод Message:makeWarning

Метод создает сообщение типа `DocumentAPI.Message.Severity_Warning` с заданным текстом.

3.23.8 Метод Message:makeError

Метод создает сообщение типа DocumentAPI.Message.Severity_Error с заданным текстом.

3.24Таблица DocumentAPI.Color

Таблица DocumentAPI.Color представляет либо цветовой объект RGBA, либо заданные цвета идентификатора темы. В качестве параметров конструктора используются таблицы DocumentAPI.ColorRGBA, DocumentAPI.ThemeColorID.

Пример:

```
local rgbaColor = DocumentAPI.Color(DocumentAPI.ColorRGBA(255, 0, 0, 255))
local themeColor = DocumentAPI.Color(DocumentAPI.ThemeColorID_Text1)
```

3.24.1 Метод Color:getRGBAColor

Метод возвращает цвет DocumentAPI.ColorRGBA.

Пример:

```
local rgbaColor = color:getRGBAColor()
if rgbaColor then
  print(rgbaColor.r, rgbaColor.g, rgbaColor.b, rgbaColor.a)
end
```

3.24.2 Метод Color:getThemeColorID

Метод возвращает цвет идентификатора темы DocumentAPI.ThemeColorID.

Пример:

```
local themeColorID = color:getThemeColorID()
if themeColorID then
  print(themeColorID)
end
```

3.24.3 Метод Color:__eq

Метод используется для определения эквивалентности двух значений цвета.

Пример:

```
if color1 == color2 then
  print("Два одинаковых цвета")
end
```

3.24.4 Метод Color:__ne

Метод используется для определения неэквивалентности двух значений цвета.

Пример:

```
if color1 == color2 then
  print("Два одинаковых цвета")
end
```

3.25 Таблица DocumentAPI.ThemeColorID

В Таблице 105 представлены типы идентификаторов цветов тем. Используется в `DocumentAPI.Color`.

Таблица 105 – Типы идентификаторов цветов тем

Наименование константы	Описание
`DocumentAPI.ThemeColorID_Background1`	Фон1
`DocumentAPI.ThemeColorID_Text1`	Текст1
`DocumentAPI.ThemeColorID_Background2`	Фон2
`DocumentAPI.ThemeColorID_Text2`	Текст2
`DocumentAPI.ThemeColorID_Dark1`	Темная1
`DocumentAPI.ThemeColorID_Dark2`	Темная2
`DocumentAPI.ThemeColorID_Light1`	Светлая1
`DocumentAPI.ThemeColorID_Light2`	Светлая2
`DocumentAPI.ThemeColorID_Accent1`	Акцент1
`DocumentAPI.ThemeColorID_Accent2`	Акцент2
`DocumentAPI.ThemeColorID_Accent3`	Акцент3
`DocumentAPI.ThemeColorID_Accent4`	Акцент4
`DocumentAPI.ThemeColorID_Accent5`	Акцент5
`DocumentAPI.ThemeColorID_Accent6`	Акцент6
`DocumentAPI.ThemeColorID_Hyperlink`	Гиперссылка
`DocumentAPI.ThemeColorID_FollowedHyperlink`	Следующая гиперссылка

3.26 Таблица DocumentAPI.PrintSettings

Таблица `DocumentAPI.PrintSettings` представляет установки, используемые при печати документов. Описание полей таблицы `DocumentAPI.PrintSettings` представлено в Таблице 106. Используется в `EditorAPI.printDocument`.

Таблица 106 – Описание полей таблицы `DocumentAPI.PrintSettings`

Поле	Тип	Описание
`DocumentAPI.PrintSettings.printerName`	string	Имя используемого принтера. Если не указано, то используется принтер по умолчанию. Если принтер с указанным именем недоступен, то возникает ошибка.
`DocumentAPI.PrintSettings.landscapeOrientation`	bool	Если значение равно `true`, то размер страницы поворачивается на 90 градусов. В настоящее время используется только для рабочих таблиц.
`DocumentAPI.PrintSettings.leftMargin`	number	Ширина левого поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
`DocumentAPI.PrintSettings.topMargin`	number	Ширина верхнего поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
`DocumentAPI.PrintSettings.rightMargin`	number	Ширина правого поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
`DocumentAPI.PrintSettings.bottomMargin`	number	Ширина нижнего поля. Размер указывается в типографских точках. В настоящее время используется только для рабочих таблиц.
`DocumentAPI.PrintSettings.parity`	`PageParity`	Выбор страниц для печати .
`DocumentAPI.PrintSettings.firstPage`	number	Номер первой страницы для печати.
`DocumentAPI.PrintSettings.lastPage`	number	Номер последней страницы для печати.
`DocumentAPI.PrintSettings.printSelection`	bool	Область печати. Значение по умолчанию `false`.
`DocumentAPI.PrintSettings.works`	`WorksheetP`	Вариант масштабирования при

Поле	Тип	Описание
heetPrinterFitType	rinterFitType	печати табличных документов .
DocumentAPI.PrintSettings.copies	number	Количество копий при печати.
DocumentAPI.PrintSettings.collateCopies	bool	Если параметр имеет значение false, то печать каждой отдельной страницы будет повторена заданное количество копий раз до начала печати следующей страницы. Если параметр имеет значение true, то все страницы печатаются до запуска печати следующей копии этих страниц. Значение по умолчанию false.

3.27 Таблица DocumentAPI.WorksheetPrinterFitType

В Таблице 107 представлены варианты масштабирования при печати табличных документов. Используется в качестве поля worksheetPrinterFitType таблицы DocumentAPI.PrintSettings.

Таблица 107 – Варианты масштабирования при печати табличных документов

Наименование константы	Описание
DocumentAPI.WorksheetPrinterFitType_ActualSize	Фактический размер
DocumentAPI.WorksheetPrinterFitType_ByPageScale	По масштабу страницы
DocumentAPI.WorksheetPrinterFitType_ByPageBreaksOnly	По разрыву страниц
DocumentAPI.WorksheetPrinterFitType_FitToPage	Вписать в страницу
DocumentAPI.WorksheetPrinterFitType_FitToWidth	Вписать по ширине
DocumentAPI.WorksheetPrinterFitType_FitToHeight	Вписать по высоте

3.28 Таблица DocumentAPI.PrintDocumentResult

В Таблице 108 представлены коды, возвращаемые после печати (см. EditorAPI.showPrintDialog()).

Таблица 108 – Коды, возвращаемые после печати

Наименование константы	Описание
DocumentAPI.PrintDocumentResult_Success	Печать прошла успешно.

Наименование константы	Описание
DocumentAPI.PrintDocumentResult_OneCopyPrinted	Напечатана только одна копия из заданных.
DocumentAPI.PrintDocumentResult_CancelPrinting	Печать была отменена.
DocumentAPI.PrintDocumentResult_NoPrinter	Принтер не найден.
DocumentAPI.PrintDocumentResult_BlankDocument	На печать отправлен пустой документ.

3.29 Таблица DocumentAPI.SizeU

Таблица DocumentAPI.SizeU представляет размер объекта в двухмерном пространстве. Описание полей таблицы DocumentAPI.SizeU представлено в Таблице 109.

Таблица 109 – Описание полей таблицы DocumentAPI.SizeU

Поле	Тип	Описание
DocumentAPI.SizeU.width	number	Ширина объекта в двухмерном пространстве.
DocumentAPI.SizeU.height	number	Высота объекта в двухмерном пространстве.

Пример:

```
local size = DocumentAPI.SizeU(2, 3)
print("width=", size.width, ", height=", size.height)  --(width = 2,0, height = 3,0)
```

3.29.1 Метод SizeU:toString

Возвращает информацию о размерах в виде строкового значения формата (width: <value>, height: <value>).

Пример:

```
local size = DocumentAPI.SizeU(2, 3)
print(size:toString())  --(width: 2.0, height: 3.0)
```

3.30 Таблица DocumentAPI.PointU

Таблица DocumentAPI.PointU представляет позицию объекта в двухмерном пространстве. Описание полей таблицы DocumentAPI.PointU представлено в Таблице 110.

Таблица 110 – Описание полей таблицы DocumentAPI.PointU

Поле	Описание
DocumentAPI.PointU.x	Позиция x

Поле	Описание
DocumentAPI.PointU.y	Позиция y

Пример:

```
local point = DocumentAPI.PointU(2, 3)
print("x=", point.x, ", y=", point.y)  --(x = 2.0, y = 3.0)
```

3.30.1 Метод PointU:toString

Возвращает информацию о позиции в виде строкового значения формата `(width: <value>, height: <value>)`.

Пример:

```
local point = DocumentAPI.PointU(2, 3)
print(point:toString())  --(x: 2.0, y: 3.0)
```

3.31 Таблица DocumentAPI.RectU

Таблица `DocumentAPI.RectU` представляет описание прямоугольной области. Описание полей таблицы `DocumentAPI.RectU` представлено в Таблице 111.

Таблица 111 – Описание полей таблицы `DocumentAPI.RectU`

Поле	Описание
DocumentAPI.RectU.topLeft	Координаты левого верхнего угла области, тип CellPosition.
DocumentAPI.RectU.rightBottom	Координаты правого нижнего угла области, тип CellPosition.

Пример:

```
local rect = DocumentAPI.RectU(2, 3, 4, 5)
print("tlx=", rect.topLeft.x, ", tly=", rect.topLeft.y, ", rbx=",
rect.bottomRight.x, ", rby=", rect.bottomRight.y)  --(tlx = 2.0, tly = 3.0,
brx = 4.0, bry = 5.0)
```

3.31.1 Метод RectU:toString

Возвращает информацию о прямоугольной области в виде строкового описания координат `[topLeft: (x: <value>, y: <value>), bottomRight: (x: <value>, y: <value>)]`.

Пример:

```
local point = DocumentAPI.RectU(2, 3, 4, 5)
print(point:toString())  --[topLeft: (x: 2.0, y: 3.0), bottomRight: (x: 4.0, y: 5.0)]
```

3.32 Таблица DocumentAPI.VectorUInt

Таблица `DocumentAPI.VectorUInt` предназначена для реализации массива данных.

Пример:

```
vector = DocumentAPI.VectorUInt(3)
vector[0] = 1
vector[1] = 13
vector[2] = 25
print(vector:size()) -- 3
```

3.32.1 Метод VectorUInt:size

Метод возвращает размер вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
vector:push_back(13)
vector:push_back(14)
print(vector:size()) -- 3
```

3.32.2 Метод VectorUInt:max_size

Метод возвращает максимальный размер вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()
print(vector:max_size())
```

3.32.3 Метод VectorUInt:empty

Метод возвращает `true`, если вектор не содержит элементов.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
print(vector:empty()) -- false
```

3.32.4 Метод VectorUInt:clear

Метод очищает содержимое вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
vector:clear()
print(vector:empty()) -- true
```

3.32.5 Метод VectorUInt:push_back

Метод добавляет элемент в конец вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
print(vector:size()) -- 1
```

3.32.6 Метод VectorUInt:pop_back

Метод удаляет последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
vector:pop_back()
print(vector:size()) -- 0
```

3.32.7 Метод VectorUInt:front

Метод возвращает первый элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
vector:push_back(13)
print(vector:front()) -- 12
```

3.32.8 Метод VectorUInt:back

Метод возвращает последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
vector:push_back(13)
print(vector:front()) -- 13
```

3.32.9 Метод VectorUInt:__getitem

Метод возвращает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorUInt()
vector:push_back(12)
vector:push_back(13)
print(vector:__getitem(0)) -- 12
print(vector:__getitem(1)) -- 13
```

3.32.10 Метод VectorUInt:__setitem

Метод устанавливает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorUInt(2)
vector:__setitem(0, 12)
vector:__setitem(1, 13)
print(vector:__getitem(0)) -- 12
print(vector:__getitem(1)) -- 13
```

3.33Таблица DocumentAPI.VectorString

Таблица `DocumentAPI.VectorString` предназначена для реализации массива строк.

Пример:

```
vector = DocumentAPI.VectorString(3)
vector[0] = "1"
vector[1] = "2"
vector[2] = "3"
print(vector:size()) -- 3
```

3.33.1 Метод VectorString:size

Метод возвращает размер вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
vector:push_back("13")
vector:push_back("14")
print(vector:size()) -- 3
```

3.33.2 Метод VectorString:max_size

Метод возвращает максимальный размер вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
print(vector:max_size())
```

3.33.3 Метод VectorString:empty

Метод возвращает `true`, если вектор не содержит элементов.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
print(vector:empty()) -- false
```

3.33.4 Метод VectorString:clear

Метод очищает содержимое вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
vector:clear()
print(vector:empty()) -- true
```

3.33.5 Метод VectorString:push_back

Метод добавляет элемент в конец вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
print(vector:size()) -- 1
```

3.33.6 Метод VectorString:pop_back

Метод удаляет последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
vector:pop_back()
print(vector:size()) -- 0
```

3.33.7 Метод VectorString:front

Метод возвращает первый элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
vector:push_back("13")
print(vector:front()) -- 12
```

3.33.8 Метод VectorString:back

Метод возвращает последний элемент вектора.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
vector:push_back("13")
print(vector:front()) -- 13
```

3.33.9 Метод VectorString:__getitem

Метод возвращает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorString()
vector:push_back("12")
vector:push_back("13")
print(vector:__getitem(0)) -- 12
print(vector:__getitem(1)) -- 13
```

3.33.10 Метод VectorString:__setitem

Метод устанавливает элемент вектора по заданному индексу.

Пример:

```
local vector = DocumentAPI.VectorString(2)
vector:__setitem(0, "12")
vector:__setitem(1, "13")
print(vector:__getitem(0)) -- 12
print(vector:__getitem(1)) -- 13
```

4 СПРАВОЧНИК ФУНКЦИЙ EDITORAPI

Глобальная таблица `EditorAPI` содержит функции доступа к внешней функциональности редактора.

4.1 Функция EditorAPI.getSelection

Функция `EditorAPI.getSelection` предоставляет доступ к выделенному фрагменту документа.

В открытом документе может быть выделен только один фрагмент.

При использовании в редакторе текста функция `EditorAPI.getSelection` возвращает `Range`, а при использовании в редакторе таблиц функция `EditorAPI.getSelection` возвращает `CellRange`.

Примеры:

Использование функции `EditorAPI.getSelection` в редакторе текста для печати выделенного фрагмента текста.

```
local text = EditorAPI.getSelection():extractText()
print(text)
```

Использование функции `EditorAPI.getSelection` в редакторе таблиц для печати значений ячеек в выделенном фрагменте таблицы.

```
for cell in EditorAPI.getSelection():enumerate() do
    print(cell:getFormattedValue())
end
```

4.2 Функция EditorAPI.setSelection

Функция `EditorAPI.setSelection` позволяет выделить фрагмент документа.

В открытом документе может быть выделен только один фрагмент.

Вызов функции для текстового документа:

```
EditorAPI.setSelection(range)
```

Где:

– `range` – выделяемый в текстовом документе фрагмент текста типа `DocumentAPI.Range`.

Вызов функции для табличного документа:

```
EditorAPI.setSelection(range)
```

Где:

– `range` – выделяемый в табличном документе фрагмент таблицы типа `DocumentAPI.CellRange`.

Примеры:

Использование функции `EditorAPI.setSelection` в текстовом документе:

```
EditorAPI.setSelection(document:getBlocks():getParagraph(0):getRange())
```

Использование функции `EditorAPI.setSelection` в табличном документе:

```
EditorAPI.setSelection(document:getBlocks():getTable(0):getCellRange("A1:E5")
)
```

4.3 Функция EditorAPI.messageBox

Функция `EditorAPI.messageBox()` выводит на экран сообщение с заданным текстом и отображением кнопки **ОК**, при этом исполнение макрокоманды приостанавливается до нажатия кнопки **ОК**.

Вызов:

```
messageBox(prompt                          :                          string)
messageBox(prompt                          :                          string)
messageBox(prompt : string, title : string)
```

Параметры:

– `prompt` – текст сообщения;

– `title` – заголовок окна сообщения.

Пример:

```
EditorAPI.messageBox(cell:getFormattedValue())
```

4.4 Функция EditorAPI.showPrintDialog

Функция `EditorAPI.showPrintDialog()` показывает стандартное окно печати редактора и распечатывает документ, если пользователь подтверждает необходимость

печати. Значения, возвращаемые функцией `EditorAPI.showPrintDialog()` перечислены в разделе `DocumentAPI.PrintDocumentResult`.

Пример:

```
printDocumentResult = EditorAPI.showPrintDialog()
print(printDocumentResult)
```

4.5 Функция EditorAPI.printDocument

Функция `EditorAPI.printDocument()` предоставляет возможность печати документа с заданными параметрами печати. Описание параметров печати представлено в разделе `DocumentAPI.PrintSettings`.

Значения, возвращаемые функцией `EditorAPI.printDocument()`, перечислены в разделе `DocumentAPI.PrintDocumentResult`.

Пример:

```
local printSettings = {}
printSettings.printSelection = true
EditorAPI.printDocument(printSettings)
```

4.6 Функция EditorAPI.isPrinterAvailable

Функция `EditorAPI.isPrinterAvailable` позволяет проверить доступность последнего использованного принтера. Возвращает `false`, если принтер недоступен.

Пример:

```
if EditorAPI.isPrinterAvailable() then
    EditorAPI.messageBox("Printer is available")
else
    EditorAPI.messageBox("Printer is not available")
end
```

5 ФУНКЦИИ ДЛЯ РАБОТЫ СО СТРОКАМИ В ФОРМАТЕ ЮНИКОД (UTF-8)

5.1 Функция utf8.upper

Функция `utf8.upper` возвращает строку в формате UTF-8, полученную из исходной строки путем преобразования в верхний регистр.

Вызов:

`utf8.upper(str)`

Параметры:

`str` - строка (`string`) в формате UTF-8

Возвращает:

`string`

5.2 Функция utf8.lower

Функция `utf8.lower` возвращает строку в формате UTF-8, полученную из исходной строки путем преобразования в нижний регистр.

Вызов:

```
utf8.lower(str)
```

Параметры:

– `str` – строка (`string`) в формате UTF-8.

Возвращает:

`string`

5.3 Функция utf8.substr

Функция `utf8.substr` возвращает подстроку в формате UTF-8, полученную из исходной строки путем выборки с заданной позиции первого символа и до позиции последнего указанного символа.

Вызов:

```
utf8.substr(str, first[, last])
```

Параметры:

– `str` – исходная строка (`string`) в формате UTF-8;

– `first` – позиция (`number`) первого символа строки для выборки.

– `last` – позиция (`number`) последнего символа строки для выборки.

Возвращает:

`string`

5.4 Функция utf8.compare

Функция `utf8.compare` возвращает результат сравнения двух строк согласно алгоритму сортировки по Юникоду.

Вызов:

```
utf8.compare(str1, str2, opt)
```

Параметры:

– `str1` – первая строка (`string`) в формате UTF-8;

– `str2` – вторая строка (`string`) в формате UTF-8;

– `opt` – параметр (`number`) учета регистра при сравнении:

– 0 – без учета регистра;
– 1 – с учетом регистра.

Возвращает:

`number` согласно алгоритму сортировки по Юникоду:

– -1 – если `str1` < `srt2`;
– 0 – если `str1` = `str2`;
– 1 – если `str1` > `str2`.

5.5 Функция utf8.islower

Функция `utf8.islower` проверяет, находится ли в нижнем регистре переданный символ или число.

Вызов:

```
utf8.islower(char)
```

Параметры:

– `char` – строка (string) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код символа в нижнем регистре.

5.6 Функция utf8.isupper

Функция `utf8.islower` проверяет, находится ли в верхнем регистре переданный символ или число.

Вызов:

```
utf8.isupper(char)
```

Параметры:

– `char` – строка (string) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код символа в верхнем регистре.

5.7 Функция utf8.isdigit

Функция `utf8.isdigit` проверяет, является ли цифровым символом переданный символ или число.

Вызов:

```
utf8.isdigit(char)
```

Параметры:

– `char` – строка (`string`) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код цифрового символа.

5.8 Функция utf8.isalpha

Функция `utf8.isalpha` проверяет, является ли буквенным символом переданный символ или число.

Вызов:

```
utf8.isalpha(char)
```

Параметры:

– `char` – строка (`string`) или число (`number`), представляющие код UTF-8.

Возвращает:

`bool` - имеет значение `true`, если передан код буквенного символа.

5.9 Функция utf8.next

Функция `utf8.next` позволяет получить байтовое смещение символа, следующего за указанным.

Вызов:

```
utf8.next(str[, offset])
```

Параметры:

– `str` – строка (`string`) в формате UTF-8;

– `offset` – байтовое смещение внутри UTF-8 строки (по умолчанию равно 1).

Возвращает:

`number` – байтовое смещение следующего символа.

6 ФУНКЦИИ ДЛЯ РАБОТЫ С РЕГУЛЯРНЫМИ ВЫРАЖЕНИЯМИ

6.1 Функция Re.create

Функция `Re.create` компилирует регулярное выражение и возвращает его в виде объекта. По умолчанию используется Perl - совместимый формат регулярных выражений.

Вызов:

```
Re.create(pattern)
```

Параметры:

– `pattern` (`string`) - строка шаблона.

Возвращает:

– `regex (object)` - объект Regex, который содержит скомпилированное регулярное выражение для дальнейшего использования;
– `err (string)` - сообщение об ошибке или `nil`.

6.2 Функция Re.match

Сопоставляет скомпилированное регулярное выражение с заданной исходной строкой. Возвращает найденные подстроки.

Вызов:

```
Re.match(subject, matchFlags, pattern)
```

Параметры:

– `subject (string)` – исходная строка;
– `matchFlags (int)` – флаги, задающие правила применения регулярного выражения;
– `pattern (string, Regex)` – строка шаблона или скомпилированный шаблон.

Возвращает:

– `matches (object)` – подстроки, найденные в соответствии с шаблоном;
– `err (string)` - сообщение об ошибке или `nil`.

6.2.1 Флаги, используемые в Re.match

Эти флаги определены в пространстве имен Re.Match. Они используются во всех алгоритмах. Когда регулярное выражение применяется к последовательности символов, применяются правила, описанные в Таблице 112

Таблица 112 – Описание флагов `Re.Match`

Флаг	Описание
`Default`	Указывает, что работа с регулярными выражениями происходит в соответствии с обычными правилами: `ECMA-262, спецификация языка ECMAScript, глава 15, часть 10, Регулярные Выражения.`
`NotBOB`	Указывает, что выражения ”\A” и "\`" не должны совпадать с подмножеством *[first, first)*.
`NotEOB`	Указывает, что выражения "\`", "\ z" и "\Z" не должны совпадать с подмножеством *[last, last)*.
`NotBOL`	Указывает, что выражение "^" не должны совпадать с подмножеством *[first, first)*.
`NotEOL`	Указывает, что выражение "$" не должно совпадать с подмножеством *[last, last)*.

Флаг	Описание
`NotBOW`	Указывает, что выражения "\<" и "\b" не должны совпадать с подмножеством *[first, first)*.
`NotEOW`	Указывает, что выражения "\>" и "\b" не должны совпадать с подмножеством *[last, last)*.
`Any`	Указывает, что если существует более одного совпадения, то любое из совпадений является приемлемым результатом. Будет применено самое первое встретившееся совпадение, при этом, не всегда самое точное. Используйте этот флаг, если для вас важна скорость обработки, но не особо важно качество результата.
`NotNull`	Указывает, что выражение не может быть использовано для пустой последовательности.
`Continious`	Указывает, что выражение должно применяться к подмножеству, которое начинается с начала.
`Partial`	Указывает, что если не найдено ни одного совпадения, то допустимо вернуть совпадение *[from, last)*, при этом *from! = last*, если может существовать более длинная последовательность символов *[from, to)*, из которых *[from, last)* - это префикс, который приведет к полному совпадению. Этот флаг используется при сопоставлении неполных или очень длинных текстов; дополнительную информацию см. документацию по частичным сравнениям.
`Extra`	Дает указание механизму поиска сохранять всю доступную информацию о наличии совпадений; если совпадение повторяется снова, то информация о каждом из них будет доступна через методы `match_results::captures()` или `sub_match_captures()`.
`SingleLine`	Эквивалентно обратному модификатору "m/" языка Perl; предотвращает поиск ^ после встроенного символа новой строки (для того, чтобы он совпадал только в начале исходного текста) и $ от поиска перед встроенным символом новой строки (чтобы он совпадал только в конце исходного текста).
`PrevAvail`	Указывает, что `--first` является допустимой позицией итератора, когда этот флаг установлен, тогда флаги `match_not_bol` и `match_not_bow` игнорируются алгоритмами регулярных выражений (RE.7) и итераторов (RE.8).
`DotNewLine`	Указывает, что выражение "." не распознается как символ новой строки. Это инверсия модификатора "s/" языка Perl.
`NotDotNull`	Указывает, что выражение "." не распознается как символ null ('\0').
`Posix`	Указывает, что выражение должно быть обработано в соответствии с правилом POSIX *"leftmost-longest"*, независимо от того, какое выражение было скомпилировано. Эти правила плохо работают со многими специфичными для Perl моментами, например, такими как ленивые (*"non-greedy"*) повторы.
`NoSubs`	Заставляет выражение вести себя так, как будто оно не имеет найденных подмножеств, независимо от того, сколько их присутствует на самом деле. Класс `Matches` будет содержать только информацию об общем совпадении, а не о совпадении в подмножествах.

6.3 Функция Re.search

Ищет скомпилированное регулярное выражение по заданной строке. Метод возвращает найденные подстроки.

Вызов:

```
Re.search(subject, matchFlags, pattern)
```

Параметры:

- `subject (string)` – исходная строка;
- `matchFlags (int)` – флаги, задающие правила применения регулярного выражения;
- `pattern (string, Regex)` – строка шаблона или скомпилированный шаблон.

Возвращает:

- `matches (object)` – подстроки, найденные в соответствии с шаблоном;
- `err (string)` - сообщение об ошибке или `nil`

6.4 Функция Re.replace

Находит в заданной строке все фрагменты, удовлетворяющие регулярному выражению. Каждый найденный фрагмент форматируется в соответствии с форматтером и заменяет собой исходный текст.

Вызов:

```
Re.replace(subject, formatter, matchFlags, pattern)
```

Параметры:

- `subject (string)` – исходная строка для поиска;
- `formatter (string)` – строка, задающая форматирование найденных фрагментов;
- `matchFlags (int)` – флаги, задающие правила применения регулярного выражения, а также флаги, специфичные для замены;
- `pattern (string, Regex)` – строка шаблона или скомпилированный шаблон.

Возвращает:

- `newString (string)` – новая строка с замененными подстроками;
- `err (string)` - сообщение об ошибке или `nil`.

6.5 Флаги, используемые для замены

Эти флаги определены в пространстве имен `Re.Replace`. Они используются в алгоритме, используемом методом `Re.replace()` и находятся в Таблице 113

Таблица 113 – Описание флагов для функции `Re.replace()`

Флаг	Описание
`FormatDefault`	Когда к исходному тексту применяется регулярное выражение, и находится очередной фрагмент для замены, новая строка формируется с использованием функции замены *ECMAScript*, ECMA-262, Спецификация языка ECMAScript, глава 15, часть 5.4.11 String.prototype.replace. Эта функциональность идентична описанной в руководстве Perl Format String Syntax. После того, как все неперекрывающиеся вхождения регулярного выражения найдены и заменены, фрагменты исходного текста, не соответствующие регулярному выражению, копируются в результирующую строку без изменений.
`FormatSed`	В случае, когда регулярное выражение применяется для замены в строке, новая строка формируется с использованием правил, описанных в стандарте IEEE Std 1003.1-2001, Portable Operating SystemInterface (POSIX), Shells and Utilities. См. также Sed Format String Syntax.
`FormatPerl`	В случае, когда регулярное выражение применяется для замены в строке, новая строка формируется по правилам Perl 5.
`FormatLiteral`	В случае, когда регулярное выражение применяется для замены в строке, новая строка будет являться строковой копией заменяемого текста.
`FormatNoCopy`	В случае, когда регулярное выражение применяется для замены в строке, фрагменты, не соответствующие регулярному выражению, не будут копироваться в результирующую строку.
`FormatFirstOnly`	Когда данный флаг установлен при операции поиска или замены, то заменяется только первое вхождение регулярного выражения.
`FormatAll`	Все синтаксические расширения включены, включая условные замены (*?ddexpression1:expression2*). Для дополнительных деталей см. Руководство по форматированию строк Boost.

7 КЛАСС MATCHES

Класс `Matches` содержит результат функций `Re.match()` и `Re.search()`.

7.1 Метод getFirst

Вызов:

```
position, err = matches:getFirst(group)
```

Параметры:

- `group (int, string)` – позиция (или имя группы) найденных результатов, начинающаяся с 1.

Возвращает:

- `position (int)` – первая позиция (в байтах) исходной строки;
- `err (string)` - сообщение об ошибке или `nil`.

7.2 Метод getLength

Вызов:

```
position, err = matches:getLength(group)
```

Параметры:

- `group (int, string)` – позиция (или имя группы) найденных результатов, начинающаяся с 1.

Возвращает:

- `length (int)` – длина исходной строки в байтах;
- `err (string)` - сообщение об ошибке или `nil`.

7.3 Метод getSize

Вызов:

```
size, err = matches:getSize()
```

Возвращает:

- `size (int)` – количество найденных групп;
- `err (string)` - сообщение об ошибке или `nil`.

7.4 Метод getString

Вызов:

```
substr, err = matches:getString(group, subject)
```

Параметры:

- `group (int, string)` – позиция (или имя группы) найденных результатов, начинающаяся с 1;
- `subject (string)` – исходная строка. **Внимание**: объект `Matches` сохраняет только смещения и не хранит исходную строку. Таким образом, необходимо передать ту же строку, которая использовалась для поиска.

Возвращает:

- `substr (string)` – найденная подстрока;

– `err (string)` - сообщение об ошибке или `nil`.

7.5 Метод _tostring

Стандартная метафункция.

`string = matches:__tostring()`

Пример:

```
local str = "-Номер:1234"
local regex = Re.create("-(\\w+):(\\d{4})")

local matches, err = Re.match(str, Re.Match.Default, regex)
print(tostring(regex), tostring(matches))

local number = matches:getString(3, str)
print(number)
```